

Package ‘XINA’

July 5, 2018

Type Package

Title Multiplexes isobaric mass tagged-based kinetics data for network analysis

Version 1.0.0

Date 2018-07-05

Author Lee, Lang Ho and Singh, Sasha A.

Maintainer

Lee, Lang Ho <lhlee@bwh.harvard.edu> and Singh, Sasha A. <sasingh@bwh.harvard.edu>

Description An intuitive R package simplifies network analyses output from multiplexed high-dimensional proteomics/transcriptomics kinetics data

Copyright This software script is protected by the Copyright Act of 1976, 17 U.S.C. §§ 101-810, as amended. Rights reserved. Please contact The Brigham and Women’s Hospital, Inc. for further information.

Encoding UTF-8

LazyData true

License GPL v3.0

Depends R (>= 3.5), png, tools, plyr, alluvial, ggplot2, RColorBrewer, graphics, gridExtra, mclust, stringr, igraph, STRINGdb

Imports mclust, stringr, plyr, circlize, alluvial, ggplot2, RColorBrewer, graphics, gridExtra, igraph, STRINGdb, Biobase

biocViews SystemsBiology, Proteomics, RNA-Seq, Protein-protein interaction network

RoxygenNote 6.0.1

R topics documented:

add_legend	2
alluvial_enriched	3
alluvial_enrichment_tests	4
calculate_centrality_scores	5
default_size	6
desc	6
draw_alluvial_plot	7
draw_circular_plot	8
example_clusters	8
extract_data_column	9
find_similar_clusters	9

generate_count_table	10
generate_superset	10
get_colors	11
get_color_for_nodes	11
get_comigrations_by_name	12
get_condition_biased_comigrations	13
get_layout	14
get_mTOR_proteins	14
get_random_data	15
get_stats	16
get_theme_blank	16
get_unknown_ppi_nodes	17
gn	18
hprd_ppi	18
length2	19
load_previous_results	19
make_random_xina_data	20
mutate_colors	21
organize_clusters	22
plot_clusters	23
plot_clusters_all	24
plot_condition_compositions	25
plot_enrichment_results	26
plot_NA	27
rank_centrality	27
xina_analysis	28
xina_clustering	29
xina_enrichment	30
xina_plot_all	32
xina_plot_bycluster	34
xina_plot_single	36

Index	39
--------------	-----------

add_legend	<i>add_legend</i>
------------	-------------------

Description

Add plot legend and locate it outside of a network plot

Usage

```
add_legend(legend_location = "bottomright", ...)
```

Arguments

legend_location	
...	Network centrality score matrix Numeric, complex, or logical vectors.

alluvial_enriched *alluvial_enriched*

Description

'alluvial_enriched' draws an alluvial plot and finds comigrated proteins. The comigration is a group of proteins that show the same expression pattern, classified and evaluated by XINA clustering, in at least two conditions. XINA can reduce the dataset complexity by filtering based on the number of comigrated proteins (size, 'comigration_size' parameter) and perform an enrichment test (P-value of Fisher's exact test, 'pval_threshold') to determine significance of enriched comigrations. The Fisher's exact test can only be done for two conditions at a time. The following 2x2 table was used to calculate the P-value from the Fisher's exact test. To evaluate significance of co-migrated proteins from cluster #1 in control to cluster #2 in test group,

	-	cluster #1 in control	other clusters in control
cluster #2 in test		65 (TP)	175 (FP)
other clusters in test		35 (FN)	979 (TN)

Usage

```
alluvial_enriched(clustering_result, selected_conditions,
  comigration_size = 0, pval_threshold = 1, pval_method = "fdr",
  cex = 0.7, alpha = 0.3)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
selected_conditions	A vector of condition names used in XINA clustering results. The number of selected conditions should be at least two.
comigration_size	The number of proteins comigrated together in the selected conditions of XINA clustering results. Default is 0
pval_threshold	This option is available only when you selected two conditions for comigration search.
pval_method	Method for p-value adjustment. See p.adjust
cex	Scaling of fonts of category labels. Default if 0.7. See alluvial
alpha	Transparency of the stripes. Default if 0.3. See alluvial

Value

A data frame containing comigrations and an alluvial plot showing comigrations

Examples

```
## Not run:
library(XINA)
```

```

# load XINA example data
data(xina_example)
example_clusters

# Get the experimental conditions in the example data
classes <- as.vector(example_clusters$condition)

# Get comigrations without any thresholds
all_comigrations <- alluvial_enriched(example_clusters, classes)

# Get comigrations that have >= 5 size (the number of comigrated proteins)
all_cor_enriched <- alluvial_enriched(example_clusters, classes, comigration_size=5)

# Get all the comigrations between Control and Stimulus1
comigrations_Control_Stimulus1 <- alluvial_enriched(example_clusters,
c(classes[1],classes[2]))

# Get comigrations between Control and Stimulus1, that have >=5 size
comigrations_Control_Stimulus1_over5 <- alluvial_enriched(example_clusters,
c(classes[1],classes[2]), comigration_size=5)

# Get comigrations between Control and Stimulus1,
# that have >= 5 size and enrichment FDR <= 0.01
comigrations_Control_Stimulus1_pval0.01_size5 <- alluvial_enriched(example_clusters,
c(classes[1],classes[2]), comigration_size=5, pval_threshold=0.01)

# Get comigrations between Control and Stimulus1,
# that have >= 5 size and enrichment Benjamini & Yekutieli <= 0.01
comigrations_Control_Stimulus1_BY0.01_size5 <- alluvial_enriched(example_clusters,
c(classes[1],classes[2]), comigration_size=5, pval_threshold=0.01, pval_method="BY")

## End(Not run)

```

alluvial_enrichment_tests

alluvial_enrichment_tests

Description

Fisher's exact test to calculate the significance over all comigrations. The following 2x2 table was used to calculate p-value from Fisher's exact test. To evaluate significance of comigrated proteins from cluster #1 in control to cluster #2 in test condition,

	cluster #1 in control	other clusters in control
cluster #2 in test	65 (TP)	175 (FP)
other clusters in test	35 (FN)	979 (TN)

'alluvial_enrichment_tests' also provides another statistical methods including Hypergeometric test and Chi-square test.

Usage

```
alluvial_enrichment_tests(count_table, c1, c2, non_cluster = 0,
  test_type = "fisher")
```

Arguments

count_table	A data frame generated by using count .
c1	A selected cluster in the first condition.
c2	A selected cluster in the second condition.
non_cluster	The cluster number for proteins that were not detected in a specific sample. Default is 0.
test_type	Enrichment test type. 'fisher' = Fisher's exact test, 'hyper' = Hypergeometric test, 'chisq' = Chi-square test

Value

P-value of comigration enrichment test and 2x2 table information

```
calculate_centrality_scores
      calculate_centrality_scores
```

Description

'calculate_centrality_scores' computes network centrality scores

Usage

```
calculate_centrality_scores(net, centrality_type = "Degree")
```

Arguments

net	protein-protein interaction network of igraph
centrality_type	the maximum number of clusters

Value

A vector of network centrality scores

default_size	<i>default_size</i>
--------------	---------------------

Description

Calculate image size based on the number of clusters

Usage

```
default_size(max_cluster)
```

Arguments

max_cluster the maximum number of clusters

Value

A vector of plot width and height

desc	<i>A character vector containing 19,396 human gene descriptions This is for the randome data generation of XINA desc</i>
------	--

Description

- Human gene description corresponding to 'gn' vector

Usage

```
desc
```

Format

A character vector containing 19,396 human gene descriptions

Source

<https://www.ncbi.nlm.nih.gov/gene>

draw_alluvial_plot	<i>draw_alluvial_plot</i>
--------------------	---------------------------

Description

'draw_alluvial_plot' draw a alluvial plot

Usage

```
draw_alluvial_plot(clustering_result, selected_conditions, count_table,
  alluvia_colors = NULL, cex = 0.7, alpha = 0.3)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering .
selected_conditions	A vector of condition names used in XINA clustering results. The number of selected conditions should be at least two.
count_table	A data frame generated by using count .
alluvia_colors	A vector containing the user-defined colors for each alluvium.
cex	Size of cluster number on block axis. Default if 0.7. See alluvial .
alpha	Transparency of alluvia colors. Default is 0.3. See alluvial .

Value

An alluvial plot displaying comigrations and the data frame containing the input count_table with colors.

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)
example_clusters

# get a vector of experimental conditions analyzed in the clustering results
classes <- as.vector(example_clusters$condition)

comigrations_size_over5 <- alluvial_enriched(example_clusters, classes, comigration_size=5)
draw_alluvial_plot(example_clusters, classes, comigrations_size_over5)

## End(Not run)
```

draw_circular_plot	<i>draw_circular_plot</i>
--------------------	---------------------------

Description

draw_circular_plot draw a circular plot

Usage

```
draw_circular_plot(clustering_result, selected_conditions, count_table)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
selected_conditions	A vector of condition names used in XINA clustering results. The number of selected conditions should be at least two.
count_table	A data frame generated by using count

Value

A circular plot displaying comigrations

example_clusters	<i>Randomly generated example datasets for XINA users. A dataset containing the XINA clustering results. example_clusters</i>
------------------	---

Description

- aligned. XINA clustering results aligned by conditions
- data_column. Column names for data matrix
- out_dir. Not available in this example dataset
- nClusters. The number of user-desired clusters. It's 30 in the example.
- max_cluster. The number of clusters found in the dataset. It's 21 in the example.
- chosen_model. The chosen covariance model for the example dataset. It's VEI in the example
- optimal_BIC. BIC at the optimized clustering. It's 29473.57 in the example
- condition. The experimental conditions in the dataset.
- color_for_condition. The default color for the conditions that will be used in XINA plot drawing.
- color_for_clusters. The default color for the clusters that will be used in XINA clustering plot.
- norm_method. The used normalization method to standardize the input data. It's "sum_normalization" in the example.

Usage

```
example_clusters
```


Format

A list with the example XINA clustering result

extract_data_column	<i>extract_data_column</i>
---------------------	----------------------------

Description

Extract data column names from XINA clustering result

Usage

```
extract_data_column(col_head_of_clustering)
```

Arguments

col_head_of_clustering
Column names of XINA clustering result

Value

A vector containing column names of data matrix

find_similar_clusters	<i>find_similar_clusters</i>
-----------------------	------------------------------

Description

Compare clusters and find similar ones

Usage

```
find_similar_clusters(clustering_result, threshold = 0.95)
```

Arguments

clustering_result
A list containing XINA clustering results. See [xina_clustering](#)

threshold
Pearson's r threshold to find similar ones

Value

Write a csv file containing similar clustering information based on the given Pearson's R threshold

generate_count_table	<i>generate_count_table</i>
----------------------	-----------------------------

Description

Count the number of comigrated proteins using [count](#)

Usage

```
generate_count_table(clustering_result, selected_conditions, comigration_size)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
selected_conditions	A vector of condition names used in XINA clustering results.
comigration_size	The number of proteins comigrated together in the selected conditions of XINA clustering results. Default is 0.

Value

A data frame containing comigrations.

generate_superset	<i>generate_superset</i>
-------------------	--------------------------

Description

Merge input kinetics files

Usage

```
generate_superset(f_names, data_column, delim = ",",
  norm = "sum_normalization")
```

Arguments

f_names	A vector of .csv file paths containing kinetics data
data_column	A vector of column names containing data matrix
delim	The delimiter of input file (default is ',')
norm	The normalization method. It should be one of c('sum_normalization', 'zscore'). Default is 'sum_normalization'.

Value

A data frame containing kinetics data obtained from files in the f_names vector

get_colors	<i>get_colors</i>
------------	-------------------

Description

Generate color series for XINA graphics

Usage

```
get_colors(nClusters, set = "", colorset = NULL)
```

Arguments

nClusters	The number of clusters
set	Pre-defined color series set
colorset	manually defined color codes

Value

A vector for color code of XINA graphics

get_color_for_nodes	<i>get_color_for_nodes</i>
---------------------	----------------------------

Description

Pre-defined 30 colors

Usage

```
get_color_for_nodes()
```

Value

A vector for color code of XINA graphics

```
get_comigrations_by_name
      get_comigrations_by_name
```

Description

'get_comigrations_by_name' finds proteins comigrated with the given proteins

Usage

```
get_comigrations_by_name(clustering_result, selected_conditions, protein_list,
  cex = 0.7, alpha = 0.3)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
selected_conditions	A vector of condition names used in XINA clustering results. The number of selected conditions should be at least two.
protein_list	A vector containing gene names.
cex	Size of cluster number on block axis. Default if 0.7. See alluvial
alpha	Transparency of alluvia colors. Default is 0.3. See alluvial

Value

An alluvial plot displaying comigrations and the data frame containing comigrations of the input proteins

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)

# the clustering result table
all_proteins <- as.character(example_clusters$aligned$`Gene name`)
# get a vector of experimental conditions analyzed in the clustering results
classes <- as.vector(example_clusters$condition)

comigrated_prots_all <- get_comigrations_by_name(example_clusters, classes, all_proteins[1:3])

## End(Not run)
```

```
get_condition_biased_comigrations
      get_condition_biased_comigrations
```

Description

get comigrations that at least one biased cluster is involved in. Biased clusters are defined by

Usage

```
get_condition_biased_comigrations(clustering_result, count_table = NULL,
  selected_conditions, condition_composition, threshold_percent = 50,
  color_for_null = "gray", color_for_highly_matched = "red4", cex = 0.7,
  alpha = 0.3)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
count_table	A data frame generated by using count . If count_table is NULL (by default), XINA will consider all the comigrations.
selected_conditions	A vector of condition names used in XINA clustering results. The number of selected conditions should be at least two.
condition_composition	The resulting data frame of 'plot_condition_compositions'. See plot_condition_compositions .
threshold_percent	Default is 50. The percentage threshold for finding condition-biased clusters
color_for_null	A color for non-condition-biased comigrations. Default is 'gray'
color_for_highly_matched	A color for comigrations that are involved with more than two condition-biased clusters. Default is 'red4'
cex	Size of cluster number on block axis. Default if 0.7. See alluvial .
alpha	Transparency of alluvia colors. Default is 0.3. See alluvial .

Value

An alluvial plot displaying comigrations and the data frame containing condition-biased comigrations.

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)

# get a vector of experimental conditions analyzed in the clustering results
conditions <- as.vector(example_clusters$condition)
```

```
# get condition composition information
condition_composition <- plot_condition_compositions(example_clusters)

comigrations_size10 <- alluvial_enriched(example_clusters, conditions, comigration_size=10)
# Finding condition-biased comigrations by 50% threshold
condition_biased_comigrations <-
get_condition_biased_comigrations(clustering_result=example_clusters,
count_table=comigrations_size10, selected_conditions=conditions,
condition_composition=condition_composition)

# Finding condition-biased comigrations by 70% threshold
condition_biased_comigrations <-
get_condition_biased_comigrations(clustering_result=example_clusters,
count_table=comigrations_size10, selected_conditions=conditions,
condition_composition=condition_composition,
threshold_percent=70)

## End(Not run)
```

get_layout

get_layout

Description

Get igraph layout by the number of nodes

Usage

```
get_layout(subnet_condition)
```

Arguments

subnet_condition
A igraph sub-network

Value

igraph network layout

get_mTOR_proteins

get_mTOR_proteins

Description

Get mTOR pathway genes

Usage

```
get_mTOR_proteins(time_points, conditions)
```

Arguments

time_points	A vector containing time points of the data matrix
conditions	A vector containing condition information, for example normal, disease and drug treated disease.

Value

A vector containing mTOR pathway gene names

get_random_data	<i>get_random_data</i>
-----------------	------------------------

Description

Get randomized time-series data

Usage

```
get_random_data(time_points, conditions, num_total, percent.sign = 0.1,  
  equal = TRUE)
```

Arguments

time_points	A vector containing time points of the data matrix
conditions	A vector containing condition information, for example normal, disease and drug treated disease.
num_total	The number of total proteins to be generated
percent.sign	Percentage of differentially expressed proteins. Ignored when equal=FALSE.
equal	If equal is TRUE, all the conditions will have numbers between 0 and 1. If it is FALSE, the first three conditions will have different ranges. First condition will have numbers from 0.3 to 0.4. Second condition will have numbers from 0.6 to 0.8. Third condition will have numbers from 0.3 to 0.5. Other conditions will have numbers from 0 to 1.

Value

A list containing randomly generated data matrix

get_stats	<i>get_stats</i>
-----------	------------------

Description

Calculate statistics of the given data for XINA network analysis

Usage

```
get_stats(centrality_results, na.rm = FALSE)
```

Arguments

centrality_results	Network centrality score data frame calculated by XINA network module
na.rm	If it is FALSE, no exclusion of NA values.

Value

A data frame containing statistics of XINA network centrality scores

get_theme_blank	<i>get_theme_blank</i>
-----------------	------------------------

Description

Predefined ggplot theme for removing ticks, titles and labels of X and Y axis

Usage

```
get_theme_blank()
```

Value

A ggplot theme

`get_unknown_ppi_nodes` *get_unknown_ppi_nodes*

Description

Get proteins with no known interactions within the cluster based on the used protein-protein interaction database source

Usage

```
get_unknown_ppi_nodes(xina_result, cl)
```

Arguments

`xina_result` A list containing XINA network analysis results. See [xina_analysis](#)
`cl` the clustering number of XINA clustering results. See [xina_clustering](#)

Value

A data frame containing proteins with no known interactions within the cluster based on the used protein-protein interaction database source

Examples

```
## Not run:
library(XINA)
library(STRINGdb)

# load XINA example data
data(xina_example)

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version='10', species=9606, score_threshold=0, input_directory='' )
string_db

# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db, flag_simplify=FALSE)

# Extract unknown PPI nodes in the cluster #1
get_unknown_ppi_nodes(xina_result, 1)

## End(Not run)
```

gn	<i>A character vector containing 19,396 human genes This is for the random data generation of XINA gn</i>
----	---

Description

- Characters of human genes

Usage

gn

Format

A character vector containing 19,396 human genes

Source

<https://www.ncbi.nlm.nih.gov/gene>

hprd_ppi	<i>Protein-protein interaction resource downloaded from HPRD DB A data frame containing HRPD protein-protein interaction data hprd_ppi</i>
----------	--

Description

- gene_symbol_1. Gene name interacting with gene name in 'gene_symbol_2'
- gene_symbol_2. Gene name interacting with gene name in 'gene_symbol_1'
- Experiment_type. Experimental or computational methods supporting the interaction

Usage

hprd_ppi

Format

A data frame containing HRPD protein-protein interaction data

Source

<http://www.hprd.org/>

length2	<i>length2</i>
---------	----------------

Description

Customized function for vector length calculation

Usage

```
length2(x, na.rm = FALSE)
```

Arguments

x	A vector
na.rm	If it is FALSE, no exclusion of NA values.

Value

A vector length

load_previous_results	<i>load_previous_results</i>
-----------------------	------------------------------

Description

Get previous XINA clustering results to R space

Usage

```
load_previous_results(clustering_dir = getwd(), data_column = NULL,
  fp_clusters = "xina_clusters.csv")
```

Arguments

clustering_dir	The directory path of XINA clustering results
data_column	A vector containing column names of data matrix
fp_clusters	File path of XINA clustering results

Value

Comma-separated file containing aligned XINA clustering results.

Examples

```
## Not run:
library(XINA)

# Generate random multiplexed time-series data
random_data_info <- make_random_xina_data()
# 'nClusters' is the number of desired clusters.
# 'mclust' will chose the most optimized number of clusters
# considering Bayesian information criteria (BIC). BIC of 'mclust' is negative BIC,
# thus the higher BIC, the more optimized clustering.
# Data files
data_files <- paste(random_data_info$conditions, ".csv", sep='')

# time points of the data matrix
data_column <- random_data_info$time_points

# mclust requires the fixed random seed to get reproduce the clustering results
set.seed(0)

# Run the model-based clustering
example_clusters <- xina_clustering(data_files, data_column=data_column, nClusters=30)

# Reload the clustering result
example_clusters_reloaded <- load_previous_results(".")
head(example_clusters_reloaded$aligned)

## End(Not run)
```

make_random_xina_data *make_random_xina_data*

Description

Generate random proteomics dataset for testing XINA 'make_random_xina_data' will make random proteomics data for XINA test. The generated data will have three conditions and seven time points, c("0hr", "2hr", "6hr", "12hr", "24hr", "48hr", "72hr").

Usage

```
make_random_xina_data(n = 500, mtor = TRUE, time_points = c("0hr", "2hr",
  "6hr", "12hr", "24hr", "48hr", "72hr"), conditions = c("Control",
  "Stimulus1", "Stimulus2"))
```

Arguments

n	The number of proteins for one condition. Default is 500.
mtor	If it is TRUE (default), mTOR pathway genes will be significant. If it is FALSE, randomly selected genes will be significant in first three conditions.
time_points	A vector containing time points of the data matrix
conditions	A vector containing condition information, for example normal, disease and drug treated disease.

Value

Three comma-separated files containing time-series data for XINA

Examples

```
## Not run:
library(XINA)
make_random_xina_data()
g1 <- read.csv("Control.csv", check.names=FALSE, stringsAsFactors = FALSE)
head(g1)
g2 <- read.csv("Stimulus1.csv", check.names=FALSE, stringsAsFactors = FALSE)
head(g2)
g3 <- read.csv("Stimulus2.csv", check.names=FALSE, stringsAsFactors = FALSE)
head(g3)

# Model-based clustering
example_clusters <- xina_clustering(c("Control.csv", "Stimulus1.csv", "Stimulus2.csv"),
  data_column=c("0hr", "6hr", "12hr", "24hr", "48hr", "72hr"), nClusters=30)
head(example_clusters$aligned)

# Plot clustered plots
plot_clusters(example_clusters)

# Plot pie-charts of condition composition
condition_composition <- plot_condition_compositions(example_clusters)

## End(Not run)
```

mutate_colors

mutate_colors

Description

'mutate_colors' generates new color scheme for XINA clustering plot based on condition composition results ([plot_condition_compositions](#)). If any clusters have higher percentage than the 'threshold_percent', XINA will assign new colors in accordance to 'color_for_condition'. If not, XINA will give 'gray' color or user-defined color via 'null_color' parameter.

Usage

```
mutate_colors(condition_composition, color_for_condition, null_color = "gray",
  threshold_percent = 50)
```

Arguments

condition_composition	A data frame generated by plot_condition_compositions
color_for_condition	A vector like 'color_for_condition' of xina_clustering
null_color	Default is 'gray'. This color is for clusters that are not biased to any of experimental conditions
threshold_percent	Default is 50. The percentage threshold for giving new colors

Value

A data frame containing statistics of XINA network centrality scores

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)
example_clusters

# Plot condition composition pie-chart with default option
condition_composition <- plot_condition_compositions(example_clusters)
example_clusters$color_for_clusters <- mutate_colors(condition_composition,
example_clusters$color_for_condition)
plot_clusters(example_clusters, xval=c(0,2,6,12,24,48,72), xylab=FALSE)

## End(Not run)
```

organize_clusters	<i>organize_clusters</i>
-------------------	--------------------------

Description

Compare clusters and find similar ones

Usage

```
organize_clusters(clustering_dir = getwd(), super_ds, file_out = TRUE)
```

Arguments

clustering_dir	The directory path of XINA clustering results
super_ds	XINA clusters
file_out	If it is TRUE, it writes the aligned clustering informaion to "xina_clusters_aligned.csv" file.

Value

Comma-separated file containing aligned XINA clustering results.

plot_clusters	<i>plot_clusters</i>
---------------	----------------------

Description

Draw all the clustering results. 'plot_clusters' draws two plots, scaled and unscaled line graphs. Scaled graphs have same y limits that are 0 to 1 by default, but can be changed via 'y_lim' parameter.

Usage

```
plot_clusters(clustering_result, y_lim = NULL, xval = NULL, xylab = TRUE,
             ggplot_theme = NULL)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
y_lim	Y axis limit. If you set y_lim=c(0,1), 'plot_clusters' will plot line graphs scaled from 0 to 1 in y-axis Default is NULL, which means unscaled line graphs.
xval	Change X axis values and labels. Default is data_column of the clustering result list
xylab	If it is FALSE, x and y labels will be blank. If it is TRUE (default), x and y labels will be shown.
ggplot_theme	This is ggplot theme to modify XINA clustering plot.

Value

Line graphs of all the clusters

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)
example_clusters

# Draw clustering plots
plot_clusters(example_clusters)

# Apply theme to the clustering plot
theme1 <- theme(title=element_text(size=8, face='bold'),
               axis.text.x = element_text(size=7),
               axis.text.y = element_blank(),
               axis.ticks.x = element_blank(),
               axis.ticks.y = element_blank(),
               axis.title.x = element_blank(),
               axis.title.y = element_blank())
plot_clusters(example_clusters, ggplot_theme=theme1)

## End(Not run)
```

plot_clusters_all	<i>plot_clusters_all</i>
-------------------	--------------------------

Description

Draw line graphs of all the proteins in the given dataset

Usage

```
plot_clusters_all(clustering_result, selected_condition = NULL)
```

Arguments

clustering_result
A list containing XINA clustering results. See [xina_clustering](#)

selected_condition
A condition name to draw the kinetics plot

Value

a list containing clustering results and pdf file containing a BIC plot in current working directory.

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)
example_clusters

# Plot kinetics of all the proteins in Control
plot_clusters_all(example_clusters, selected_condition="Control")

# Plot kinetics of all the proteins in Stimulus1
plot_clusters_all(example_clusters, selected_condition="Stimulus1")

# Plot kinetics of all the proteins in Stimulus2
plot_clusters_all(example_clusters, selected_condition="Stimulus2")

# Plot kinetics of all the proteins in three data
plot_clusters_all(example_clusters)

## End(Not run)
```

```
plot_condition_compositions
      plot_condition_compositions
```

Description

computes condition composition of the XINA clustering results and draws pie-charts.

Usage

```
plot_condition_compositions(clustering_result, bullseye = FALSE,
  ggplot_theme = NULL)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
bullseye	If it is TRUE, draw bullseye plot instead of the pie-chart. Default is FALSE
ggplot_theme	This is ggplot theme to modify condition composition pie-chart and bulles eye plots.

Value

A condition composition plot and a data frame containing condition compositions of the clusters

Examples

```
## Not run:
library(XINA)

# load XINA example data
data(xina_example)

# Plot condition composition pie-chart with default option
plot_condition_compositions(example_clusters)

# Make a new color code for conditions
condition_colors <- c("tomato","steelblue1","gold")
names(condition_colors) <- example_clusters$condition
example_clusters$color_for_condition <- condition_colors

# Draw condition composition pie-chart with the new color code
plot_condition_compositions(example_clusters)

# Draw condition composition bullseye plot
plot_condition_compositions(example_clusters, bullseye = TRUE)

## End(Not run)

# Save the clustering plot as a PNG file
png(file="condition_composition.png", width=4000, height=4000, res=300)
plot_condition_compositions(example_clusters)
dev.off()
```

```
# Add ggplot theme to modify the condition composition plot
theme_template <- theme(legend.key.size = unit(0.3, "cm"),
  legend.text=element_text(size=5), title=element_text(size=7, face='bold'))
condition_composition <- plot_condition_compositions(example_clusters,
  ggplot_theme=theme_template)
```

plot_enrichment_results

plot_enrichment_results

Description

Plot GO and KEGG enrichment results

Usage

```
plot_enrichment_results(enriched_results,
  term_description = "term_description", sig_score = "pvalue",
  num_terms = 0, get_log = TRUE)
```

Arguments

enriched_results	GO or KEGG enrichment results. See xina_enrichment and xina_enrichment
term_description	Description of terms to be drawn on Y axis. Default is "term_description" of XINA enrichment results.
sig_score	significant score to plot on X axis. Default is "pvalue".
num_terms	The number of terms to be plotted. Default is 0, which means no limit.
get_log	If this is TRUE, 'plot_enrichment_results' will take -log10 of p-values.

Value

ggplot bar graph

Examples

```
## Not run:
library(XINA)
library(Biobase)
library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version="10", species=9606,
  score_threshold=0, input_directory="" )
string_db
```

```
# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

# Enrichment test and get significantly enriched functional terms
# that have adjusted p-value less than 0.1
kegg_enriched <- xina_enrichment(string_db, protein_list,
  enrichment_type = "KEGG", pval_threshold=0.1)
plot_enrichment_results(kegg_enriched$KEGG, num_terms=10)

## End(Not run)
```

plot_NA	<i>plot_NA</i>
---------	----------------

Description

Draw NULL plot

Usage

```
plot_NA()
```

Value

a empty plot

rank_centrality	<i>rank_centrality</i>
-----------------	------------------------

Description

Give ranks based on network centrality scores

Usage

```
rank_centrality(centrality_score, type, num_breaks = 5)
```

Arguments

centrality_score	
	Network centrality score matrix
type	Network centrality score type, such as 'Eigenvector'
num_breaks	The number of ranks

Value

A vector containing ranks

xina_analysis	<i>xina_analysis</i>
---------------	----------------------

Description

xina_analysis is to analyze protein-protein interaction(PPI) networks using STRINGdb and igraph R package. This module computes PPI networks within each XINA clusters.

Usage

```
xina_analysis(clustering_result, ppi_db, is_stringdb = TRUE,
              flag_simplify = TRUE, node_shape = "sphere", num_clusters_in_row = 5,
              img_size = NULL, img_qual = 300)
```

Arguments

clustering_result	A list containing XINA clustering results. See xina_clustering
ppi_db	STRINGdb object
is_stringdb	If it is TRUE (default), XINA will process 'ppi_db' as STRINGdb, but it is FALSE, XINA will accepts your 'ppi_db' as it is. You can make your own igraph network using customized PPI information instead of STRINGdb.
flag_simplify	If it is TRUE (default), XINA will exclude unconnected proteins
node_shape	You can choose node shape. Default is "sphere". See shapes
num_clusters_in_row	The number of clusters in a row on the XINA network plot. Default is 5.
img_size	Set the image size. For width=1000 and height=1500, it is img_size=c(1000,1500).
img_qual	Set the image resolution. Default is 300.

Value

A PNG file (XINA_Cluster_Networks.png) displaying PPI network plots of all the clusters and a list containing XINA network analysis results.

Item	Description
All_network	PPI network of all the input proteins
Sub_network	A list containing PPI networks of each clusters
Data	XINA clustering results. See xina_clustering
Nodes	A list of proteins in each cluster
Conditions	A list of experimental condition of proteins in each cluster
Titles	A list of plot titles for XINA plotting
out_dir	A directory path storing XINA network analysis results
is_stringdb	False = different PPI DB and TRUE = STRING DB

Examples

```
## Not run:
library(XINA)
```

```

library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version="10", species=9606, score_threshold=0,
input_directory="" )
string_db

# Run XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

# Draw plot including isolated nodes
xina_result <- xina_analysis(example_clusters, string_db, flag_simplify=FALSE)

# Change node shape to rectangle and plot 10 clusters in a row and node shape
# You can choose one of c("sphere", "circle", "square", "csquare", "rectangle",
# "crectangle", "vrectangle", "pie" (see vertex.shape.pie))
xina_result <- xina_analysis(example_clusters, string_db, node_shape='rectangle',
num_clusters_in_row=10)

# Run XINA with HPRD protein-protein interaction database
data(HPRD)
net_all <- simplify(graph_from_data_frame(d=hprd_ppi, directed=F),
remove.multiple = F, remove.loops = T)
xina_result <- xina_analysis(example_clusters, net_all, is_stringdb=FALSE)

## End(Not run)

```

xina_clustering

xina_clustering

Description

Clustering multiplexed time-series omics data to find co-abundance profiles

Usage

```

xina_clustering(f_names, data_column, out_dir = getwd(), nClusters = 20,
norm = "sum_normalization", chosen_model = "")

```

Arguments

<code>f_names</code>	A vector containing input file (.csv) paths
<code>data_column</code>	A vector containing column names (1st row of the input file) of data matrix
<code>out_dir</code>	A directory path for saving clustering results. (default: <code>out_dir=getwd()</code>)
<code>nClusters</code>	The number of desired maximum clusters
<code>norm</code>	Default is "sum_normalization". Sum-normalization is to divide the data matrix by row sum. If you want to know more about sum-normalization, see https://www.ncbi.nlm.nih.gov/pubmed/19861354 . "zscore" is to calculate Z score for each protein. See scale .

chosen_model [mclustModelNames](#)

Value

a plot containing a BIC plot in current working directory and a list containing below information:

Item	Description
clusters	XINA clustering results
aligned	XINA clustering results aligned by ID
data_column	Data matrix column names
out_dir	The directory path containing XINA results
nClusters	The number of clusters desired by user
max_cluster	The number of clusters optimized by BIC
chosen_model	The used covariance model for model-based clustering
optimal_BIC	BIC of the optimized covariance model
condition	Experimental conditions of the user input data
color_for_condition	Colors assigned to each experimental conditions which is used for condition composition plot
color_for_clusters	Colors assigned to each clusters which is used for XINA clustering plot
norm_method	Used normalization method

Examples

```
## Not run:
library(XINA)

# Generate random multiplexed time-series data
random_data_info <- make_random_xina_data()
# 'nClusters' is the number of desired clusters.
# 'mclust' will chose the most optimized number of clusters
# considering Bayesian information criteria (BIC). BIC of 'mclust' is negative BIC,
# thus the higher BIC, the more optimized clustering.
# Data files
data_files <- paste(random_data_info$conditions, ".csv", sep='')

# time points of the data matrix
data_column <- random_data_info$time_points

# mclust requires the fixed random seed to get reproduce the clustering results
set.seed(0)

# Run the model-based clustering to find co-abundance profiles
example_clusters <- xina_clustering(data_files, data_column=data_column, nClusters=30)

## End(Not run)
```

xina_enrichment

xina_enrichment

Description

xina_enrichment conducts functional enrichment tests using gene ontology or KEGG pathway terms for a given protein list

Usage

```
xina_enrichment(string_db, protein_list, enrichment_type = "GO",
  pval_threshold = 0.05, methodMT = "fdr")
```

Arguments

string_db	STRINGdb object
protein_list	A vector of gene names to draw protein-protein interaction network.
enrichment_type	A functional annotation for the enrichment test. 'enrichment_type' should be one of 'GO' and 'KEGG',
pval_threshold	P-value threshold to get significantly enriched terms from the given proteins
methodMT	Method for p-value adjustment. See p.adjust . Default is 'fdr'.

Value

A list of data frames containing enrichment results

Examples

```
## Not run:
library(XINA)
library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version="10", species=9606, score_threshold=0, input_directory="" )
string_db

# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

# Select proteins that showed cluster #24 in the Stimulus2 condition
subgroup <- subset(example_clusters$aligned, Stimulus2==1)
protein_list <- as.vector(subgroup$`Gene name`)

# Enrichment test using KEGG pathway terms that have adjuseted p-value less than 0.1
kegg_enriched <- xina_enrichment(string_db, protein_list,
  enrichment_type = "KEGG", pval_threshold=0.1)
plot_enrichment_results(kegg_enriched$KEGG, num_terms=10)

# Enrichment test using GO terms that have adjuseted p-value less than 0.1
go_enriched <- xina_enrichment(string_db, protein_list,
  enrichment_type = "GO", pval_threshold=0.1)

# Plot the enriched GO cellular component terms
plot_enrichment_results(go_enriched$Component, num_terms=10)

## End(Not run)
```

xina_plot_all

xina_plot_all

Description

xina_plot_all is to draw protein-protein interaction network plots of all the clusters

Usage

```
xina_plot_all(xina_result, clustering_result, condition = "all",
              centrality_type = NULL, flag_simplify = TRUE, num_breaks = 5,
              layout_specified = "", vertex_label_flag = FALSE,
              vertex.label.color = "black", vertex.color = "", edge.color = NULL,
              vertex.label.dist = 0.6, vertex.label.cex = 0.8, edge.arrow.size = 0.4,
              vertex.size = 10, vertex.shape = "sphere", legend_location = "bottom",
              num_clusters_in_row = 5, flag_unknown_only = FALSE, img_size = NULL,
              img_qual = 300)
```

Arguments

xina_result A list containing XINA network analysis results. See [xina_analysis](#)

clustering_result A list containing XINA clustering results. See [xina_clustering](#)

condition Default is 'all', which means use all the proteins to draw graphs. If you specify the experimental condition name used for XINA clustering, xina_plot_all will draw graphs using specific condition's proteins.

centrality_type 'centrality_type' should be one of c('Degree', 'Eigenvector', 'Hub', 'Authority', 'Closeness', 'Betweenness')

Centrality score	igraph function
Degree	degree
Eigenvector	eigen_centrality
Hub	hub_score
Authority	authority_score
Closeness	closeness
Betweenness	betweenness

flag_simplify If it is TRUE (default), XINA will exclude unconnected proteins

num_breaks 'num_breaks' is the number of ranks based on network centrality. Default is 5.

layout_specified This can change network layout. 'layout_specified' should be one of c('sphere', 'star', 'gem', 'tree', 'circle', 'random', 'nicely'). XINA's layouts are based on igraph's layout. See [layout_](#)

Layout	igraph layout name
sphere	layout_on_sphere
star	layout_as_star
gem	layout_with_gem

tree	layout_as_tree
circle	layout_in_circle
random	layout_randomly
nicely	layout_nicely

Default is 'layout_nicely' of igraph

vertex_label_flag

If vertex_label_flag is TRUE (default), igraph network graphs will be labeled by gene names. If vertex_label_flag is FALSE, igraph network graphs will be drawn without labels.

vertex.label.color

Color of labels. Default is black.

vertex.color

Color of nodes. Default is pink.

edge.color

Color of edges. Default is pink.

vertex.label.dist

Distance between node and label. Default is 0.6.

vertex.label.cex

Size of labels. Default is 0.8.

edge.arrow.size

Size of edges. Default is 0.4.

vertex.size

Size of nodes. Default is 0.4.

vertex.shape

You can choose node shape. Default is 'sphere'. See [shapes](#)

legend_location

If centrality_type is chosen, xina_plot_single add the color legend guiding rank of nodes based on the centrality score. Default is 'bottomright', but you can choose one of these 'bottomright', 'bottom', 'bottomleft', 'left', 'topleft', 'top', 'topright', 'right' and 'center'.

num_clusters_in_row

The number of clusters in a row on the XINA network plot. Default is 5.

flag_unknown_only

If this is TRUE, 'xina_plot_all' will plot proteins that do not have any protein-protein interaction in the given database.

img_size

Set the image size. For width=1000 and height=1500, it is img_size=c(1000,1500). Default is c(3000,3000).

img_qual

Set the image resolution. Default is 300.

Value

PNG images of PPI network plots of all the clusters

Examples

```
## Not run:
library(XINA)
library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters
```

```
# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version='10', species=9606, score_threshold=0, input_directory='' )
string_db

# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

# XINA network plots
xina_plot_all(xina_result, example_clusters)

# XINA network plots for Control condition
xina_plot_all(xina_result, example_clusters, condition='Control')

## End(Not run)
```

xina_plot_bycluster *xina_plot_bycluster*

Description

xina_plot_bycluster is to draw protein-protein interaction network plots of each cluster

Usage

```
xina_plot_bycluster(xina_result, clustering_result, cl = NULL,
  condition = "all", flag_legend = TRUE, centrality_type = NULL,
  flag_simplify = TRUE, layout_specified = "", vertex_label_flag = TRUE,
  vertex.label.dist = 0.6, vertex.label.cex = 0.8, edge.arrow.size = 0.4,
  vertex.size = 10, vertex.shape = "sphere", vertex.color = "",
  edge.color = "darkgray", legend_location = "bottom",
  flag_unknown_only = FALSE)
```

Arguments

xina_result A list containing XINA network analysis results. See [xina_analysis](#)

clustering_result A list containing XINA clustering results. See [xina_clustering](#)

cl Cluster number in the XINA clustering results

condition Default is 'all', which means use all the proteins to draw graphs. If you specify the experimental condition name used for XINA clustering,

flag_legend If it is TRUE, a legend will be printed out together.

centrality_type 'centrality_type' should be one of c('Degree', 'Eigenvector', 'Hub', 'Authority', 'Closeness', 'Betweenness')

Centrality score	igraph function
Degree	degree
Eigenvector	eigen_centrality
Hub	hub_score

Authority	authority_score
Closeness	closeness
Betweenness	betweenness

flag_simplify If it is TRUE (default), XINA will exclude unconnected proteins

layout_specified

This can change network layout. 'layout_specified' should be one of c('sphere', 'star', 'gem', 'tree', 'circle', 'random', 'nicely'). XINA's layouts are based on igraph's layout. See [layout_](#)

Layout	igraph layout name
sphere	layout_on_sphere
star	layout_as_star
gem	layout_with_gem
tree	layout_as_tree
circle	layout_in_circle
random	layout_randomly
nicely	layout_nicely

Default is 'layout_nicely' of igraph

vertex_label_flag

If **vertex_label_flag** is TRUE (default), igraph network graphs will be labeled by gene names If **vertex_label_flag** is FALSE, igraph network graphs will be drawn without labels

vertex.label.dist

Distance between node and label. Default is 0.6

vertex.label.cex

Size of labels Default is 0.8

edge.arrow.size

Size of edges Default is 0.4

vertex.size Size of nodes Default is 0.4

vertex.shape You can choose node shape. Default is 'sphere'. See [shapes](#)

vertex.color Color of nodes. Default is pink.

edge.color Color of edges. Default is pink.

legend_location

If **centrality_type** is chosen, **xina_plot_single** add the color legend guiding rank of nodes based on the centrality score. Default is 'bottomright', but you can choose one of these 'bottomright', 'bottom', 'bottomleft', 'left', 'topleft', 'top', 'topright', 'right' and 'center'.

flag_unknown_only

If this is TRUE, 'xina_plot_bycluster' will plot proteins that do not have any protein-protein interaction in the given database

Value

A PNG file (XINA_Cluster_Networks.png) displaying protein-protein interaction network plots of all the clusters and a list containing XINA network analysis results

PNG images of PPI network plots of all the clusters

Examples

```
## Not run:
library(XINA)
library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version='10', species=9606, score_threshold=0, input_directory='' )
string_db

# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

# plot cluster #1
xina_plot_bycluster(xina_result, example_clusters, cl=1)

# plot PPI network of Control condition in cluster #1
xina_plot_bycluster(xina_result, example_clusters, cl=1, condition='Control')

## End(Not run)
```

xina_plot_single	<i>xina_plot_single</i>
------------------	-------------------------

Description

xina_plot_single draws protein-protein interaction network plot for given 'protein_list'.

Usage

```
xina_plot_single(xina_result, protein_list, centrality_type = NULL,
  layout_specified = "", vertex_label_flag = TRUE, main = NULL,
  vertex.label.color = "black", vertex.color = NA,
  edge.color = "darkgray", vertex.label.dist = 0.6,
  vertex.label.cex = 0.8, edge.arrow.size = 0.4, vertex.size = 10,
  vertex.shape = "sphere", legend_location = "bottom", num_breaks = 5,
  digits_round_up = 5, flag_simplify = TRUE, flag_legend = TRUE)
```

Arguments

- xina_result A list containing XINA network analysis results. See [xina_analysis](#)
- protein_list A vector of gene names to draw a protein-protein interaction network graph.
- centrality_type 'centrality_type' should be one of c('Degree', 'Eigenvector', 'Hub', 'Authority', 'Closeness', 'Betweenness')

Centrality score	igraph function
Degree	degree

Eigenvector	eigen_centrality
Hub	hub_score
Authority	authority_score
Closeness	closeness
Betweenness	betweenness

layout_specified

This can change network layout. 'layout_specified' should be one of c('sphere', 'star', 'gem', 'tree', 'circle', 'random', 'nicely'). XINA's layouts are based on igraph's layout. See [layout_](#)

Layout	igraph layout name
sphere	layout_on_sphere
star	layout_as_star
gem	layout_with_gem
tree	layout_as_tree
circle	layout_in_circle
random	layout_randomly
nicely	layout_nicely

Default is 'layout_nicely' of igraph

vertex_label_flag

If vertex_label_flag is TRUE (default), igraph network graphs will be labeled by gene names. If vertex_label_flag is FALSE, igraph network graphs will be drawn without labels.

main

Title of network figure. If it is NULL (default), it will be the number of plotted proteins.

vertex.label.color

Color of labels. Default is black.

vertex.color

Color of nodes. Default is pink.

edge.color

Color of edges. Default is pink.

vertex.label.dist

Distance between node and label. Default is 0.6.

vertex.label.cex

Size of labels. Default is 0.8.

edge.arrow.size

Size of edges. Default is 0.4.

vertex.size

Size of nodes. Default is 0.4.

vertex.shape

You can choose node shape. Default is 'sphere'. See [shapes](#).

legend_location

If centrality_type is chosen, 'xina_plot_single' adds the color legend guiding rank of nodes based on the centrality score. Default is 'bottomright', but you can choose one of these 'bottomright', 'bottom', 'bottomleft', 'left', 'topleft', 'top', 'topright', 'right' and 'center'.

num_breaks

'num_breaks' is the number of ranks based on network centrality. Default is 5.

digits_round_up

See [Round](#).

flag_simplify

If it is TRUE (default), XINA will exclude unconnected proteins.

flag_legend

If it is TRUE, a legend will be printed out together.

Value

A PNG file (XINA_Cluster_Networks.png) displaying protein-protein interaction network plots of all the clusters and a list containing XINA network analysis results

Examples

```
## Not run:
library(XINA)
library(STRINGdb)

# load XINA example data
data(xina_example)
example_clusters

# Get STRING database for protein-protein intereaction information
string_db <- STRINGdb$new( version='10', species=9606, score_threshold=0, input_directory='' )
string_db

# XINA analysis with STRING DB
xina_result <- xina_analysis(example_clusters, string_db)

subgroup <- subset(example_clusters$aligned, Stimulus2==21)
protein_list <- subgroup$`Gene name`

# Calculate protein-protein interaction network
xina_plot_single(xina_result, protein_list)

# Calculate protein-protein interaction network and Eigenvector centrality
eigen_info <- xina_plot_single(xina_result, protein_list, centrality_type='Eigenvector')

## End(Not run)
```

Index

*Topic **datasets**

- desc, [6](#)
- example_clusters, [8](#)
- gn, [18](#)
- hprd_ppi, [18](#)

add_legend, [2](#)

alluvial, [3](#), [7](#), [12](#), [13](#)

alluvial_enriched, [3](#)

alluvial_enrichment_tests, [4](#)

authority_score, [32](#), [35](#), [37](#)

betweenness, [32](#), [35](#), [37](#)

calculate_centrality_scores, [5](#)

closeness, [32](#), [35](#), [37](#)

count, [5](#), [7](#), [8](#), [10](#), [13](#)

default_size, [6](#)

degree, [32](#), [34](#), [36](#)

desc, [6](#)

draw_alluvial_plot, [7](#)

draw_circular_plot, [8](#)

eigen_centrality, [32](#), [34](#), [37](#)

example_clusters, [8](#)

extract_data_column, [9](#)

find_similar_clusters, [9](#)

generate_count_table, [10](#)

generate_superset, [10](#)

get_color_for_nodes, [11](#)

get_colors, [11](#)

get_comigrations_by_name, [12](#)

get_condition_biased_comigrations, [13](#)

get_layout, [14](#)

get_mTOR_proteins, [14](#)

get_random_data, [15](#)

get_stats, [16](#)

get_theme_blank, [16](#)

get_unknown_ppi_nodes, [17](#)

gn, [18](#)

hprd_ppi, [18](#)

hub_score, [32](#), [34](#), [37](#)

layout_, [32](#), [35](#), [37](#)

layout_as_star, [32](#), [35](#), [37](#)

layout_as_tree, [33](#), [35](#), [37](#)

layout_in_circle, [33](#), [35](#), [37](#)

layout_nicely, [33](#), [35](#), [37](#)

layout_on_sphere, [32](#), [35](#), [37](#)

layout_randomly, [33](#), [35](#), [37](#)

layout_with_gem, [32](#), [35](#), [37](#)

length2, [19](#)

load_previous_results, [19](#)

make_random_xina_data, [20](#)

mclustModelNames, [30](#)

mutate_colors, [21](#)

organize_clusters, [22](#)

p.adjust, [3](#), [31](#)

plot_clusters, [23](#)

plot_clusters_all, [24](#)

plot_condition_compositions, [13](#), [21](#), [25](#)

plot_enrichment_results, [26](#)

plot_NA, [27](#)

rank_centrality, [27](#)

Round, [37](#)

scale, [29](#)

shapes, [28](#), [33](#), [35](#), [37](#)

xina_analysis, [17](#), [28](#), [32](#), [34](#), [36](#)

xina_clustering, [3](#), [7–10](#), [12](#), [13](#), [17](#), [21](#), [23–25](#), [28](#), [29](#), [32](#), [34](#)

xina_enrichment, [26](#), [30](#)

xina_plot_all, [32](#)

xina_plot_bycluster, [34](#)

xina_plot_single, [36](#)