

# Package ‘SPONGE’

November 20, 2024

**Type** Package

**Title** Sparse Partial Correlations On Gene Expression

**Version** 1.29.0

**Description** This package provides methods to efficiently detect competitive endogenous RNA interactions between two genes. Such interactions are mediated by one or several miRNAs such that both gene and miRNA expression data for a larger number of samples is needed as input. The SPONGE package now also includes spongEffects: ceRNA modules offer patient-specific insights into the miRNA regulatory landscape.

**License** GPL (>=3)

**LazyData** TRUE

**LazyDataCompression** xz

**RoxygenNote** 7.1.2

**Depends** R (>= 3.6)

**Suggests** testthat, knitr, rmarkdown, visNetwork, ggrepel, gridExtra, digest, doParallel, bigmemory, GSVA

**Imports** Biobase, stats, ppcor, logging, foreach, doRNG, data.table, MASS, expm, gRbase, glmnet, igraph, iterators, tidyverse, caret, dplyr, biomaRt, randomForest, ggridges, cvms, ComplexHeatmap, ggplot2, MetBrewer, rlang, tnet, ggpubr, stringr, tidyr

**VignetteBuilder** knitr

**biocViews** GeneExpression, Transcription, GeneRegulation, NetworkInference, Transcriptomics, SystemsBiology, Regression, RandomForest, MachineLearning

**git\_url** <https://git.bioconductor.org/packages/SPONGE>

**git\_branch** devel

**git\_last\_commit** 9357494

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2024-11-19

**Author** Markus List [aut, cre] (ORCID: <<https://orcid.org/0000-0002-0941-4168>>),  
 Markus Hoffmann [aut] (ORCID: <<https://orcid.org/0000-0002-1920-288X>>),  
 Lena Strasser [aut] (ORCID: <<https://orcid.org/0009-0007-7881-6818>>)

**Maintainer** Markus List <markus.list@tum.de>

## Contents

build_classifier_central_genes . . . . .	3
calibrate_model . . . . .	4
ceRNA_interactions . . . . .	5
check_and_convert_expression_data . . . . .	6
define_modules . . . . .	6
enrichment_modules . . . . .	7
ensembl.df . . . . .	8
filter_ceRNA_network . . . . .	8
fn_combined_centrality . . . . .	9
fn_discretize_spongeeffects . . . . .	10
fn_elasticnet . . . . .	10
fn_exact_match_summary . . . . .	11
fn_filter_network . . . . .	11
fn_gene_miRNA_F_test . . . . .	12
fn_get_model_coef . . . . .	12
fn_get_rss . . . . .	13
fn_get_semi_random_OE . . . . .	13
fn_get_shared_miRNAs . . . . .	14
fn_OE_module . . . . .	14
fn_RF_classifier . . . . .	15
fn_weighted_degree . . . . .	16
genes_pairwise_combinations . . . . .	16
gene_expr . . . . .	17
get_central_modules . . . . .	17
mircode_ensg . . . . .	18
mircode_symbol . . . . .	18
mir_expr . . . . .	19
mir_interactions . . . . .	19
plot_accuracy_sensitivity_specificity . . . . .	19
plot_confusion_matrices . . . . .	20
plot_density_scores . . . . .	21
plot_heatmaps . . . . .	22
plot_involved_miRNAs_to_modules . . . . .	23
plot_top_modules . . . . .	25
precomputed_cov_matrices . . . . .	25
precomputed_null_model . . . . .	26
prepare_metabric_for_spongEffects . . . . .	26
prepare_tcga_for_spongEffects . . . . .	27
Random_spongEffects . . . . .	28
sample_zero_mscor_cov . . . . .	29

sample_zero_mscor_data . . . . .	30
sponge . . . . .	31
sponge_build_null_model . . . . .	33
sponge_compute_p_values . . . . .	34
sponge_edge_centralities . . . . .	35
sponge_gene_miRNA_interaction_filter . . . . .	35
sponge_network . . . . .	38
sponge_node_centralities . . . . .	39
sponge_plot_network . . . . .	39
sponge_plot_network_centralities . . . . .	40
sponge_plot_simulation_results . . . . .	41
sponge_run_benchmark . . . . .	41
sponge_subsampling . . . . .	43
targetscan_ensg . . . . .	44
targetscan_symbol . . . . .	44
test_cancer_gene_expr . . . . .	45
test_cancer_metadata . . . . .	45
test_cancer_mir_expr . . . . .	45
train_cancer_gene_expr . . . . .	46
train_cancer_metadata . . . . .	46
train_cancer_mir_expr . . . . .	46
train_ceRNA_interactions . . . . .	47
train_network_centralities . . . . .	47

**Index** **48**

`build_classifier_central_genes`  
*build classifiers for central genes*

**Description**

build classifiers for central genes

**Usage**

```
build_classifier_central_genes(
  train_gene_expr,
  test_gene_expr,
  train_enrichment_modules,
  test_enrichment_modules,
  train_meta_data,
  test_meta_data,
  train_meta_data_type = "TCGA",
  test_meta_data_type = "TCGA",
  metric = "Exact_match",
  tunegrid_c = c(1:100),
  n.folds = 10,
```

```

    repetitions = 3
  )

```

### Arguments

**train\_gene\_expr** expression data of train dataset, genenames must be in rownames  
**test\_gene\_expr** expression data of test dataset, genenames must be in rownames  
**train\_enrichment\_modules** return of enrichment\_modules()  
**test\_enrichment\_modules** return of enrichment\_modules()  
**train\_meta\_data** meta data of train dataset  
**test\_meta\_data** meta data of test dataset  
**train\_meta\_data\_type** TCGA or METABRIC  
**test\_meta\_data\_type** TCGA or METABRIC  
**metric** metric (Exact\_match, Accuracy) (default: Exact\_match)  
**tunegrid\_c** defines the grid for the hyperparameter optimization during cross validation (caret package) (default: 1:100)  
**n.folds** number of folds to be calculated  
**repetitions** number of k-fold cv iterations (default: 3)

### Value

model for central genes

---

calibrate_model	<i>tests and trains a model for a disease using a training and test data set (e.g., TCGA-BRCA and METABRIC)</i>
-----------------	---

---

### Description

tests and trains a model for a disease using a training and test data set (e.g., TCGA-BRCA and METABRIC)

**Usage**

```
calibrate_model(
  Input,
  modules_metadata,
  label,
  sampleIDs,
  Metric = "Exact_match",
  tune_grid_c = c(1:100),
  n_folds = 10,
  repetitions = 3
)
```

**Arguments**

Input	Features to use for model calibration.
modules_metadata	metadata table containing information about samples/patients
label	Column of metadata to use as label in classification model
sampleIDs	Column of metadata containing sample/patient IDs to be matched with column names of spongEffects scores
Metric	metric (Exact_match, Accuracy) (default: Exact_match)
tune_grid_c	defines the grid for the hyperparameter optimization during cross validation (caret package) (default: 1:100)
n_folds	number of folds (default: 10)
repetitions	number of k-fold cv iterations (default: 3)
modules	return from enrichment_modules() function

**Value**

returns a list with the trained model and the prediction results Calibrate classification RF classification model

returns a list with the trained model and the prediction results

---

ceRNA\_interactions      *ceRNA interactions*

---

**Description**

ceRNA interactions

**Usage**

```
ceRNA_interactions
```

**Format**

A data table of ceRNA interactions typically provided by sponge

---

check\_and\_convert\_expression\_data

*Checks if expression data is in matrix or ExpressionSet format and converts the latter to a standard matrix. Alternatively, a big.matrix descriptor object can be supplied to make use of shared memory between parallelized workers through the bigmemory package.*

---

### Description

Checks if expression data is in matrix or ExpressionSet format and converts the latter to a standard matrix. Alternatively, a big.matrix descriptor object can be supplied to make use of shared memory between parallelized workers through the bigmemory package.

### Usage

```
check_and_convert_expression_data(expr_data)
```

### Arguments

expr\_data      expr\_data as matrix or ExpressionSet

### Value

expr\_data as matrix

### Examples

```
## Not run: check_and_convert_expression_data(gene_expr)
```

---

define\_modules      *Functions to define Sponge modules, created as all the first neighbors of the most central genes*

---

### Description

Functions to define Sponge modules, created as all the first neighbors of the most central genes

### Usage

```
define_modules(  
  network,  
  central.modules = F,  
  remove.central = T,  
  set.parallel = T  
)
```

**Arguments**

network	Network as dataframe and list of central nodes. First two columns of the dataframe should contain the information of the nodes connected by edges.
central.modules	consider central gene as part of the module (default: False)
remove.central	Possibility of keeping or removing (default) central genes in the modules (default: T)
set.parallel	paralleling calculation of define_modules() (default: F)

**Value**

List of modules. Module names are the corresponding central genes.

---

enrichment_modules	<i>Calculate enrichment scores</i>
--------------------	------------------------------------

---

**Description**

Calculate enrichment scores

**Usage**

```
enrichment_modules(  
  Expr.matrix,  
  modules,  
  bin.size = 100,  
  min.size = 10,  
  max.size = 200,  
  min.expr = 10,  
  method = "OE",  
  cores = 1  
)
```

**Arguments**

Expr.matrix	ceRNA expression matrix
modules	Result of define_modules()
bin.size	bin size (default: 100)
min.size	minimum module size (default: 10)
max.size	maximum module size (default: 200)
min.expr	minimum expression (default: 10)
method	Enrichment to be used (Overall Enrichment: OE or Gene Set Variation Analysis: GSVA) (default: OE)
cores	number of cores to be used to calculate enrichment scores with gsva or ssgea methods. Default 1

**Value**

matrix containing module enrichment scores (module x samples)

---

ensembl.df	<i>example potential central nodes</i>
------------	--

---

**Description**

example potential central nodes

**Usage**

ensembl.df

**Format**

(downloaded via biomaRt)

---

filter_ceRNA_network	<i>prepare ceRNA network and network centralities from SPONGE / SPONGEdb for spongEffects</i>
----------------------	---

---

**Description**

prepare ceRNA network and network centralities from SPONGE / SPONGEdb for spongEffects

**Usage**

```
filter_ceRNA_network(
  sponge_effects,
  Node_Centrality = NA,
  add_weighted_centrality = T,
  mscor.threshold = NA,
  padj.threshold = NA
)
```

**Arguments**

`sponge_effects` the ceRNA network downloaded as R object from SPONGEdb (Hoffmann et al., 2021) or created by SPONGE (List et al., 2019) (ends with `_sponge_results` in the SPONGE vignette)



**Node\_Centrality**

the network analysis downloaded as R object from SPONGEdb (Hoffmann et al., 2021) or created by SPONGE and containing centrality measures. (List et al., 2019) (ends with `_networkAnalysis` in the SPONGE vignette, you can also use your own network centrality measurements) if `network_analysis` is NA then the function only filters the ceRNA network, otherwise it will filter the given network centralities, but will not recalculate them based on the filtered ceRNA network.

**add\_weighted\_centrality**

calculate and add weighted centrality measures to previously available centralities. Default = T

**mscor.threshold**

mscor threshold to be filtered (default: NA)

**padj.threshold** adjusted p-value to be filtered (default: NA)

**Value**

list of filtered ceRNA network and network centralities. You can access it with `list$ObjectName` for further `spongEffects` steps

---

**fn\_combined\_centrality**

*Function to calculate centrality scores Calculation of combined centrality scores as proposed by Del Rio et al. (2009)*

---

**Description**

Function to calculate centrality scores Calculation of combined centrality scores as proposed by Del Rio et al. (2009)

**Usage**

```
fn_combined_centrality(CentralityMeasures)
```

**Arguments**

**CentralityMeasures**

dataframe with centrality score measures as columns and samples as rows

**Value**

Vector containing combined centrality scores

---

fn\_discretize\_spongeeffects  
*discretize #' (functions taken from: Jerby-Arnon et al. 2018)*

---

**Description**

discretize #' (functions taken from: Jerby-Arnon et al. 2018)

**Usage**

```
fn_discretize_spongeeffects(v, n.cat)
```

**Arguments**

v	gene distance (defined by mother function OE module function)
n.cat	size of the bins (defined by mother function OE module function)

**Value**

discretized

---

fn\_elasticnet      *Computes an elastic net model*

---

**Description**

Computes an elastic net model

**Usage**

```
fn_elasticnet(x, y, alpha.step = 0.1)
```

**Arguments**

x	miRNA expression matrix
y	gene expression vector
alpha.step	Step size for alpha, the tuning parameter for elastic net.

**Value**

The best model, i.e. the one for which the selected alpha yielded the smallest residual sum of squares error

---

fn\_exact\_match\_summary  
*Calibrate classification method*

---

**Description**

Calibrate classification method

**Usage**

```
fn_exact_match_summary(data, lev = NULL, model = NULL)
```

**Arguments**

data	Dataframe with module scores/covariates (modules x samples) AND outcome variable
lev	(default: NULL)
model	(default: NULL)

**Value**

Model and confusion matrix in a list

---

fn\_filter\_network      *Preprocessing ceRNA network*

---

**Description**

Preprocessing ceRNA network

**Usage**

```
fn_filter_network(network, mscor.threshold = 0.1, padj.threshold = 0.01)
```

**Arguments**

network	ceRNA network as data (typically present in the outputs of sponge)
mscor.threshold	mscor threshold (default 0.1)
padj.threshold	adjusted p-value threshold (default 0.01)

**Value**

filtered ceRNA network

---

fn\_gene\_miRNA\_F\_test *Perform F test for gene-miRNA elastic net model*

---

**Description**

Perform F test for gene-miRNA elastic net model

**Usage**

```
fn_gene_miRNA_F_test(g_expr, m_expr, model, p.adj.threshold = NULL)
```

**Arguments**

g_expr	A gene expression matrix with samples in rows and genes in columns
m_expr	A miRNA expression matrix with samples in rows and genes in columns. Sample number and order has to agree with above gene expression matrix
model	A nested elastic net model to be tested
p.adj.threshold	Threshold for FDR corrected p-value

**Value**

return data frame with miRNA, fstat and adjusted p.value (BH).

---

fn\_get\_model\_coef *Extract the model coefficients from an elastic net model*

---

**Description**

Extract the model coefficients from an elastic net model

**Usage**

```
fn_get_model_coef(model)
```

**Arguments**

model	An elastic net model
-------	----------------------

**Value**

A data frame with miRNAs and coefficients

---

fn_get_rss	<i>Compute the residual sum of squares error for an elastic net model</i>
------------	---

---

**Description**

Compute the residual sum of squares error for an elastic net model

**Usage**

```
fn_get_rss(model, x, y)
```

**Arguments**

model	The elastic net model
x	The miRNA expression
y	The gene expression

**Value**

the RSS

---

fn_get_semi_random_OE	<i>Function to calculate semi random enrichment scores of modules OE (functions taken from: Jerby-Arnon et al. 2018)</i>
-----------------------	--

---

**Description**

Function to calculate semi random enrichment scores of modules OE (functions taken from: Jerby-Arnon et al. 2018)

**Usage**

```
fn_get_semi_random_OE(r, genes.dist.q, b.sign, num.rounds = 1000)
```

**Arguments**

r	expression matrix
genes.dist.q	values of the genes after binning (result of binning)
b.sign	does the signature contain less than 2 genes? (controll parameter) (is set by mother function (OE module function))
num.rounds	number of rounds (default: 1000)

**Value**

random signature scores

---

fn\_get\_shared\_miRNAs *Identify miRNAs for which both genes have miRNA binding sites aka miRNA response elements in the competing endogenous RNA hypothesis*

---

### Description

Identify miRNAs for which both genes have miRNA binding sites aka miRNA response elements in the competing endogenous RNA hypothesis

### Usage

```
fn_get_shared_miRNAs(geneA, geneB, mir_interactions)
```

### Arguments

geneA	The first gene
geneB	The second gene
mir_interactions	A named list of genes, where for each gene all miRNA interacting partners are listed

### Value

A vector with shared RNAs of the two genes.

---

fn\_OE\_module *Function to calculate enrichment scores of modules OE (functions taken from: Jerby-Arnon et al. 2018)*

---

### Description

Function to calculate enrichment scores of modules OE (functions taken from: Jerby-Arnon et al. 2018)

### Usage

```
fn_OE_module(
  NormCount,
  gene.sign,
  bin.size = 100,
  num.rounds = 1000,
  set_seed = 42
)
```

**Arguments**

NormCount	normalized counts
gene.sign	significant genes
bin.size	bin size (default: 100)
num.rounds	number of rounds (default: 1000)
set_seed	seed size (default: 42)

**Value**

Signature scores

---

fn_RF_classifier	<i>RF classification model</i>
------------------	--------------------------------

---

**Description**

RF classification model

**Usage**

```
fn_RF_classifier(
  Input.object,
  K,
  rep,
  metric = "Exact_match",
  tunegrid,
  set_seed = 42
)
```

**Arguments**

Input.object	data.frame made by predictors and dependent variable
K	number of folds (k-fold)
rep	number of times repeating the cross validation
metric	metric (Exact_match, Accuracy) (default: Exact_match)
tunegrid	defines the grid for the hyperparameter optimization during cross validation (caret package)
set_seed	set seed (default: 42)

---

fn_weighted_degree	<i>Function to calculate centrality scores Calculation of weighted degree scores based on Opsahl et al. (2010) Hyperparameter to tune: Alpha = 0 -&gt; degree centrality as defined in Freeman, 1978 (number of edges).</i>
--------------------	---

---

**Description**

Function to calculate centrality scores Calculation of weighted degree scores based on Opsahl et al. (2010) Hyperparameter to tune: Alpha = 0 -> degree centrality as defined in Freeman, 1978 (number of edges).

**Usage**

```
fn_weighted_degree(network, undirected = T, Alpha = 1)
```

**Arguments**

network	Network formatted as a dataframe with three columns containing respectively node1, node2 and weights
undirected	directionality of the network (default: T)
Alpha	degree centrality as defined in Barrat et al., 2004 (default: 1)

**Value**

Dataframe containing information about nodes and their weighted centrality measure

---

genes_pairwise_combinations	<i>Compute all pairwise interactions for a number of genes as indices</i>
-----------------------------	---

---

**Description**

Compute all pairwise interactions for a number of genes as indices

**Usage**

```
genes_pairwise_combinations(number.of.genes)
```

**Arguments**

number.of.genes	Number of genes for which all pairwise interactions are needed
-----------------	--

**Value**

data frame with one row per unique pairwise combination. To be used as input for the sponge method.



---

gene_expr	<i>Gene expression test data set</i>
-----------	--------------------------------------

---

**Description**

Gene expression test data set

**Usage**

```
gene_expr
```

**Format**

A data frame of expression values with samples in columns and genes in rows

---

get_central_modules	<i>prepare ceRNA network and network centralities from SPONGE / SPONGEdb</i>
---------------------	--

---

**Description**

prepare ceRNA network and network centralities from SPONGE / SPONGEdb

**Usage**

```
get_central_modules(
  central_nodes,
  node_centrality,
  ceRNA_class = c("lncRNA", "circRNA", "protein_coding"),
  centrality_measure = "Weighted_Degree",
  cutoff = 1000
)
```

**Arguments**

**central\_nodes** Vector containing Ensemble IDs of the chosen RNAs to use as central nodes for the modules.

**node\_centrality** output from `filter_ceRNA_network()` or own measurement, if own measurement taken, please provide `node_centrality_column`

**ceRNA\_class** default `c("lncRNA", "circRNA", "protein_coding")` (see <http://www.ensembl.org/info/genome/genebuild/b>)

**centrality\_measure** Type of centrality measure to use. (Default: "Weighted\_Degree", calculated in `filter_ceRNA_network()`)

**cutoff** the top cutoff modules will be returned (default: 1000)

**Value**

top cutoff modules, with selected RNAs as central genes

---

mircode_ensg	<i>mircode predicted miRNA gene interactions</i>
--------------	--

---

**Description**

mircode predicted miRNA gene interactions

**Usage**

mircode\_ensg

**Format**

A matrix gene ensembl ids vs miRNA family names.  $\geq 1$  if interaction is predicted, 0 otherwise

**Source**

<http://www.mircode.org/download.php>

---

mircode_symbol	<i>mircode predicted miRNA gene interactions</i>
----------------	--

---

**Description**

mircode predicted miRNA gene interactions

**Usage**

mircode\_symbol

**Format**

A matrix gene symbols vs miRNA family names.  $\geq 1$  if interaction is predicted, 0 otherwise

**Source**

<http://www.mircode.org/download.php>

---

mir_expr	<i>miRNA expression test data set</i>
----------	---------------------------------------

---

**Description**

miRNA expression test data set

**Usage**

mir\_expr

**Format**

A data frame of expression values with samples in columns and miRNA in rows

---

mir_interactions	<i>miRNA / gene interactions</i>
------------------	----------------------------------

---

**Description**

miRNA / gene interactions

**Usage**

mir\_interactions

**Format**

A data frame of regression coefficients typically provided by `sponge_gene_miRNA_interaction_filter`

---

plot_accuracy_sensitivity_specificity	<i>list of plots for (1) accuracy and (2) sensitivity + specificity (see Boniolo and Hoffmann 2022 et al. Fig. 3a and Fig. 3b)</i>
---------------------------------------	--

---

**Description**

list of plots for (1) accuracy and (2) sensitivity + specificity (see Boniolo and Hoffmann 2022 et al. Fig. 3a and Fig. 3b)

**Usage**

```
plot_accuracy_sensitivity_specificity(
  trained_model,
  central_genes_model = NA,
  all_expression_model = NA,
  random_model,
  training_dataset_name = "TCGA",
  testing_dataset_name = "TCGA",
  subtypes
)
```

**Arguments**

`trained_model` returned from `train_and_test_model`

`central_genes_model`  
returned from `build_classifier_central_genes()`

`all_expression_model`  
training and testing like `central_genes_model` but on ALL common expression data

`random_model` returned from `train_and_test_model` using the randomization

`training_dataset_name`  
name of training (e.g., TCGA)

`testing_dataset_name`  
name of testing set (e.g., METABRIC)

`subtypes` array of subtypes (e.g., `c("Normal", "LumA", "LumB", "Her2", "Basal")`)

**Value**

list of plots for (1) accuracy and (2) sensitivity + specificity

---

`plot_confusion_matrices`

*plots the confusion matrix from `spongEffects train_and_test()` (see Boniolo and Hoffmann 2022 et al. Fig. 3a and Fig. 3b)*

---

**Description**

plots the confusion matrix from `spongEffects train_and_test()` (see Boniolo and Hoffmann 2022 et al. Fig. 3a and Fig. 3b)

**Usage**

```
plot_confusion_matrices(trained_model, subtypes.testing.factors)
```



---

plot_heatmaps	<i>plots the heatmaps from training_and_test_model (see Boniolo and Hoffmann 2022 et al. Fig. 6)</i>
---------------	--

---

### Description

plots the heatmaps from training\_and\_test\_model (see Boniolo and Hoffmann 2022 et al. Fig. 6)

### Usage

```
plot_heatmaps(  
  trained_model,  
  spongEffects,  
  meta_data,  
  label,  
  sampleIDs,  
  Modules_to_Plot = 2,  
  show.rownames = F,  
  show.colnames = F  
)
```

### Arguments

trained_model	returned from train_and_test_model
spongEffects	output of enrichment_modules()
meta_data	metadata of samples (retrieved from prepare_tcga_for_spongEffects() or from prepare_metabric_for_spongEffects())
label	Column of metadata to use as label in classification model
sampleIDs	Column of metadata containing sample/patient IDs to be matched with column names of spongEffects scores
Modules_to_Plot	Number of modules to plot in the heatmap. Default = 2
show.rownames	Add row names (i.e. module names) to the heatmap. Default = F
show.colnames	Add column names (i.e. sample names) to the heatmap. Default = F

### Value

ComplexHeatmap object NOT FUNCTIONAL

---

```
plot_involved_miRNAs_to_modules
```

*plots the heatmap of miRNAs involved in the interactions of the modules (see Boniolo and Hoffmann 2022 et al. Fig. 7a)*

---

## Description

plots the heatmap of miRNAs involved in the interactions of the modules (see Boniolo and Hoffmann 2022 et al. Fig. 7a)

## Usage

```
plot_involved_miRNAs_to_modules(
  sponge_modules,
  trained_model,
  gene_mirna_candidates,
  k_modules = 25,
  filter_miRNAs = 3,
  bioMart_gene_symbol_columns = "hgnc_symbol",
  bioMart_gene_ensembl = "hsapiens_gene_ensembl",
  width = 5,
  length = 5,
  show_row_names = T,
  show_column_names = T,
  show_annotation_column = F,
  title = "Frequency",
  legend_height = 1.5,
  labels_gp_fontsize = 8,
  title_gp_fontsize = 8,
  legend_width = 3,
  column_title = "Module",
  row_title = "miRNA",
  row_title_gp_fontsize = 10,
  column_title_gp_fontsize = 10,
  row_names_gp_fontsize = 7,
  column_names_gp_fontsize = 7,
  column_names_rot = 45,
  unit = "cm"
)
```

## Arguments

`sponge_modules` result of `define_modules()`  
`trained_model` returned from `train_and_test_model`  
`gene_mirna_candidates`  
 output of SPONGE or SPONGEdb (miRNAs\_significance)

<code>k_modules</code>	top k modules to be shown (default: 25)
<code>filter_miRNAs</code>	min rowsum to be reach of miRNAs (default: 3.0)
<code>bioMart_gene_symbol_columns</code>	bioMart dataset column for gene symbols (e.g. human: <code>hgnc_symbol</code> , mouse: <code>mgc_symbol</code> ) (default: <code>hgnc_symbol</code> )
<code>bioMart_gene_ensembl</code>	bioMart gene ensemble name (e.g., <code>hsapiens_gene_ensembl</code> )
<code>width</code>	the width of the heatmap (default: 5)
<code>length</code>	the length of the heatmap (default: 5)
<code>show_row_names</code>	show row names (default: T)
<code>show_column_names</code>	show column names (default: T)
<code>show_annotation_column</code>	add annotation column to columns (default: F)
<code>title</code>	the title of the plot (default: "Frequency")
<code>legend_height</code>	the height of the legend (default: 1.5)
<code>labels_gp_fontsize</code>	the font size of the labels (default: 8)
<code>title_gp_fontsize</code>	the font size of the title (default: 8)
<code>legend_width</code>	the width of the legend (default: 3)
<code>column_title</code>	the column title (default: "Module")
<code>row_title</code>	the title of the rows (default: "miRNA")
<code>row_title_gp_fontsize</code>	the font size of the row title (default: 10)
<code>column_title_gp_fontsize</code>	the font size of the column title (default: 10)
<code>row_names_gp_fontsize</code>	the font size of the row names (default: 7)
<code>column_names_gp_fontsize</code>	the font size of the column names (default: 7)
<code>column_names_rot</code>	the rotation angel of the column names (default: 45)
<code>unit</code>	either cm or inch (see <code>ComplexHeatmap</code> parameter)

**Value**

plot object



---

plot_top_modules	<i>plots the top x gini index modules (see Boniolo and Hoffmann 2022 et al. Figure 5)</i>
------------------	---

---

### Description

plots the top x gini index modules (see Boniolo and Hoffmann 2022 et al. Figure 5)

### Usage

```
plot_top_modules(  
  trained_model,  
  k_modules = 25,  
  k_modules_red = 10,  
  text_size = 16  
)
```

### Arguments

trained_model	returned from train_and_test_model
k_modules	top k modules to be shown (default: 25)
k_modules_red	top k modules shown in red - NOTE: must be smaller than k_modules (default: 10)
text_size	text size (default 16)
bioMart_gene_symbol_columns	bioMart dataset column for gene symbols (e.g. human: hgnc_symbol, mouse: mgi_symbol) (default: hgnc_symbol)
bioMart_gene_ensembl	bioMart gene ensemble name (e.g., hsapiens_gene_ensembl).

### Value

plot object for lollipop plot

---

precomputed_cov_matrices	<i>covariance matrices under the null hypothesis that sensitivity correlation is zero</i>
--------------------------	---

---

### Description

covariance matrices under the null hypothesis that sensitivity correlation is zero

**Usage**

```
precomputed_cov_matrices
```

**Format**

A list (different gene-gene correlations  $k$ ) of lists (different number of miRNAs  $m$ ) of covariance matrices

---

```
precomputed_null_model
```

*A null model for testing purposes*

---

**Description**

A null model for testing purposes

**Usage**

```
precomputed_null_model
```

**Format**

A list (different gene-gene correlations  $k$ ) of lists (different number of miRNAs  $m$ ) of sampled mscor values (100 each, computed from 100 samples)

---

```
prepare_metabric_for_spongEffects
```

*prepare METABRIC formats for spongEffects*

---

**Description**

prepare METABRIC formats for spongEffects

**Usage**

```
prepare_metabric_for_spongEffects(  
  metabric_expression,  
  metabric_metadata,  
  subtypes_of_interest,  
  bioMart_gene_ensembl = "hsapiens_gene_ensembl",  
  bioMart_gene_symbol_columns = "hgnc_symbol"  
)
```

**Arguments**

metabric\_expression  
 filepath to expression data in metabric format

metabric\_metadata  
 filepath to metabric metadata in metabric format

subtypes\_of\_interest  
 array e.g., c("LumA", "LumB", "Her2", "Basal", "Normal")

bioMart\_gene\_ensembl  
 bioMart gene ensemble name (e.g., hsapiens\_gene\_ensembl). (See <https://www.bioconductor.org/packages>)  
 (default: hsapiens\_gene\_ensembl)

bioMart\_gene\_symbol\_columns  
 bioMart dataset column for gene symbols (e.g. human: hgnc\_symbol, mouse: mgi\_symbol) (default: hgnc\_symbol)

**Value**

list with metabric expression and metadata. You can access it with list\$ObjectName for further spongEffects steps

---

prepare\_tcga\_for\_spongEffects  
*prepare TCGA formats for spongEffects*

---

**Description**

prepare TCGA formats for spongEffects

**Usage**

```
prepare_tcga_for_spongEffects(
  tcga_cancer_symbol,
  normal_ceRNA_expression_data,
  tumor_ceRNA_expression_data,
  normal_metadata,
  tumor_metadata,
  clinical_data,
  tumor_stages_of_interest,
  subtypes_of_interest
)
```

**Arguments**

tcga\_cancer\_symbol  
 e.g., BRCA for breast cancer

normal\_ceRNA\_expression\_data  
 normal ceRNA expression data (same structure as input for SPONGE)

```
tumor_ceRNA_expression_data
    tumor ceRNA expression data (same structure as input for SPONGE)
normal_metadata
    metadata for normal samples (TCGA format style, needs to include column:
    sampleID, PATIENT_ID)
tumor_metadata
    metadata for tumor samples (TCGA format style, needs to include column: sam-
    pleID, PATIENT_ID)
clinical_data
    clinical data for all patients (TCGA format style, needs to include column: PA-
    TIENT_ID, AJCC_PATHOLOGIC_TUMOR_STAGE)
tumor_stages_of_interest
    array e.g., c(STAGE I', 'STAGE IA', 'STAGE IB', 'STAGE II', 'STAGE IIA')
subtypes_of_interest
    array e.g., c("LumA", "LumB", "Her2", "Basal", "Normal")
```

**Value**

list of prepared data. You can access it with `list$objectname` for further `spongEffects` steps

---

Random\_spongEffects    *build random classifiers*

---

**Description**

build random classifiers

**Usage**

```
Random_spongEffects(
  sponge_modules,
  gene_expr,
  min.size = 10,
  bin.size = 100,
  max.size = 200,
  min.expression = 10,
  replace = F,
  method = "OE",
  cores = 1
)
```

**Arguments**

```
sponge_modules  result of define_modules()
gene_expr       Input expression matrix
min.size        minimum module size (default: 10)
bin.size        bin size (default: 100)
```

max.size	maximum module size (default: 200)
replace	Possibility of keeping or removing (default) central genes in the modules (default: F)
method	Enrichment to be used (Overall Enrichment: OE or Gene Set Variation Analysis: GSVA) (default: OE)
cores	number of cores to be used to calculate enrichment scores with gsva or ssgsea methods. Default 1
train_gene_expr	expression data of train dataset, genenames must be in rownames
test_gene_expr	expression data of test dataset, genenames must be in rownames
train_meta_data	meta data of train dataset
test_meta_data	meta data of test dataset
train_meta_data_type	TCGA or METABRIC
test_meta_data_type	TCGA or METABRIC
metric	metric (Exact_match, Accuracy) (default: Exact_match)
tunegrid_c	defines the grid for the hyperparameter optimization during cross validation (caret package) (default: 1:100)
n.folds	number of folds to be calculated
repetitions	number of k-fold cv iterations (default: 3)
min.expr	minimum expression (default: 10)

**Value**

randomized prediction model Define random modules  
 A list with randomly defined modules and related enrichment scores

---

sample\_zero\_mscor\_cov *Sampling zero multiple miRNA sensitivity covariance matrices*

---

**Description**

Sampling zero multiple miRNA sensitivity covariance matrices

**Usage**

```
sample_zero_mscor_cov(
  m,
  number_of_solutions,
  number_of_attempts = 1000,
  gene_gene_correlation = NULL,
  random_seed = NULL,
  log.level = "ERROR"
)
```

**Arguments**

**m** number of miRNAs, i.e. number of columns of the matrix  
**number\_of\_solutions** stop after this many instances have been samples  
**number\_of\_attempts** give up after that many attempts  
**gene\_gene\_correlation** optional, define the correlation of the first two elements, i.e. the genes.  
**random\_seed** A random seed to be used for reproducible results  
**log.level** the log level, typically set to INFO, set to DEBUG for verbose logging

**Value**

a list of covariance matrices with zero sensitivity correlation

**Examples**

```
sample_zero_mscor_cov(m = 1,
  number_of_solutions = 1,
  gene_gene_correlation = 0.5)
```

---

sample\_zero\_mscor\_data

*Sample mscor coefficients from pre-computed covariance matrices*

---

**Description**

Sample mscor coefficients from pre-computed covariance matrices

**Usage**

```
sample_zero_mscor_data(
  cov_matrices,
  number_of_samples = 100,
  number_of_datasets = 100
)
```

**Arguments**

**cov\_matrices** a list of pre-computed covariance matrices  
**number\_of\_samples** the number of samples available in the expression data  
**number\_of\_datasets** the number of mscor coefficients to be sampled from each covariance matrix

**Value**

a vector of mscor coefficients

**See Also**

sample\_zero\_mscor\_cov

**Examples**

```
#we select from the pre-computed covariance matrices in SPONGE
#100 for m = 5 miRNAs and gene-gene correlation 0.6
cov_matrices_selected <- precomputed_cov_matrices[["5"]][["0.6"]]
sample_zero_mscor_data(cov_matrices = cov_matrices_selected,
number_of_samples = 200, number_of_datasets = 10)
```

---

sponge	<i>Compute competing endogeneous RNA interactions using Sparse Partial correlations ON Gene Expression (SPONGE)</i>
--------	---

---

**Description**

Compute competing endogeneous RNA interactions using Sparse Partial correlations ON Gene Expression (SPONGE)

**Usage**

```
sponge(
  gene_expr,
  mir_expr,
  mir_interactions = NULL,
  log.level = "ERROR",
  log.every.n = 1e+05,
  log.file = NULL,
  selected.genes = NULL,
  gene.combinations = NULL,
  each.miRNA = FALSE,
  min.cor = 0.1,
  parallel.chunks = 1000,
  random_seed = NULL,
  result_as_dt = FALSE
)
```

**Arguments**

gene_expr	A gene expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
-----------	--

<code>mir_expr</code>	A miRNA expression matrix with samples in rows and features in columns. Alternatively an object of class <code>ExpressionSet</code> .
<code>mir_interactions</code>	A named list of genes, where for each gene we list all miRNA interaction partners that should be considered.
<code>log.level</code>	The log level, can be one of "info", "debug", "error"
<code>log.every.n</code>	write to the log after every n steps
<code>log.file</code>	write log to a file, particularly useful for parallelization
<code>selected.genes</code>	Operate only on a subset of genes, particularly useful for bootstrapping
<code>gene.combinations</code>	A data frame of combinations of genes to be tested. Gene names are taken from the first two columns and have to match the names used for <code>gene_expr</code>
<code>each.miRNA</code>	Whether to consider individual miRNAs or pooling them.
<code>min.cor</code>	Consider only gene pairs with a minimum correlation specified here.
<code>parallel.chunks</code>	Split into this number of tasks if parallel processing is set up. The number should be high enough to guarantee equal distribution of the work load in parallel execution. However, if the number is too large, e.g. in the worst case one chunk per computation, the overhead causes more computing time than can be saved by parallel execution. Register a parallel backend that is compatible with <code>foreach</code> to use this feature. More information can be found in the documentation of the <code>foreach</code> / <code>doParallel</code> packages.
<code>random_seed</code>	A random seed to be used for reproducible results
<code>result_as_dt</code>	whether to return results as data table or data frame

### Value

A data frame with significant gene-gene competitive endogenous RNA or 'sponge' interactions

### Examples

```
#First, extract miRNA candidates for each of the genes
#using sponge_gene_miRNA_interaction_filter. Here we use a prepared
#dataset mir_interactions.

#Second we compute ceRNA interactions for all pairwise combinations of genes
#using all miRNAs remaining after filtering through elasticnet.
ceRNA_interactions <- sponge(
  gene_expr = gene_expr,
  mir_expr = mir_expr,
  mir_interactions = mir_interactions)
```



---

`sponge_build_null_model`*Build null model for p-value computation*

---

**Description**

Build null model for p-value computation

**Usage**

```
sponge_build_null_model(  
  number_of_datasets = 1e+05,  
  number_of_samples,  
  cov_matrices = precomputed_cov_matrices,  
  ks = seq(0.2, 0.9, 0.1),  
  m_max = 8,  
  log.level = "ERROR"  
)
```

**Arguments**

<code>number_of_datasets</code>	the number of datasets defining the precision of the p-value
<code>number_of_samples</code>	the number of samples in the expression data
<code>cov_matrices</code>	pre-computed covariance matrices
<code>ks</code>	a sequence of gene-gene correlation values for which null models are computed
<code>m_max</code>	null models are build for each elt in ks for 1 to m_max miRNAs
<code>log.level</code>	The log level of the logging package

**Value**

a list (for various values of m) of lists (for various values of k) of lists of simulated data sets, drawn from a set of precomputed covariance matrices

**Examples**

```
sponge_build_null_model(100, 100,  
  cov_matrices = precomputed_cov_matrices[1:3], m_max = 3)
```

---

`sponge_compute_p_values`*Compute p-values for SPONGE interactions*

---

### Description

This method uses pre-computed covariance matrices that were created for various gene-gene correlations (0.2 to 0.9 in steps of 0.1) and number of miRNAs (between 1 and 8) under the null hypothesis that the sensitivity correlation is zero. Datasets are sampled from this null model and allow for an empirical p-value to be computed that is only significant if the sensitivity correlation is higher than can be expected by chance given the number of samples, correlation and number of miRNAs. p-values are adjusted independently for each parameter combination using Benjamini-Hochberg FDR correction.

### Usage

```
sponge_compute_p_values(sponge_result, null_model, log.level = "ERROR")
```

### Arguments

<code>sponge_result</code>	A data frame from a sponge call
<code>null_model</code>	optional, pre-computed simulated data
<code>log.level</code>	The log level of the logging package

### Value

A data frame with sponge results, now including p-values and adjusted p-value

### See Also

`sponge_build_null_model`

### Examples

```
sponge_compute_p_values(ceRNA_interactions,  
null_model = precomputed_null_model)
```

---

sponge\_edge\_centralities  
*Computes edge centralities*

---

**Description**

Computes edge betweenness centrality for the ceRNA interaction network induced by the results of the SPONGE method.

**Usage**

```
sponge_edge_centralities(sponge_result)
```

**Arguments**

sponge\_result The output generated by the sponge method.

**Value**

data table or data frame with gene, degree, eigenvector and betweenness

**See Also**

sponge

**Examples**

```
sponge_edge_centralities(ceRNA_interactions)
```

---

sponge\_gene\_miRNA\_interaction\_filter  
*Determine miRNA-gene interactions to be considered in SPONGE*

---

**Description**

The purpose of this method is to limit the number of miRNA-gene interactions we need to consider in SPONGE. There are 3 filtering steps: 1. variance filter (optional). Only consider genes and miRNAs with variance > var.threshold. 2. miRNA target database filter (optional). Use a miRNA target database provided by the user to filter for those miRNA gene interactions for which evidence exists. This can either be predicted target interactions or experimentally validated ones. 3. For each remaining interaction of a gene and its regulating miRNAs use elastic net regression to achieve a) Feature selection: We only retain miRNAs that influence gene expression b) Effect strength: The sign of the coefficients allows us to filter for miRNAs that down-regulate gene expression. Moreover, we can use the coefficients to rank the miRNAs by their relative effect strength. We strongly recommend setting up a parallel backend compatible with the foreach package. See example and the documentation of the foreach and doParallel packages.

**Usage**

```
sponge_gene_miRNA_interaction_filter(
  gene_expr,
  mir_expr,
  mir_predicted_targets,
  elastic.net = TRUE,
  log.level = "ERROR",
  log.file = NULL,
  var.threshold = NULL,
  F.test = FALSE,
  F.test.p.adj.threshold = 0.05,
  coefficient.threshold = -0.05,
  coefficient.direction = "<",
  select.non.targets = FALSE,
  random_seed = NULL,
  parallel.chunks = 100
)
```

**Arguments**

gene_expr	A gene expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
mir_expr	A miRNA expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
mir_predicted_targets	A data frame with miRNA in cols and genes in rows. A 0 indicates the miRNA is not predicted to target the gene, >0 otherwise. If this parameter is NULL all miRNA-gene interactions are tested
elastic.net	Whether to apply elastic net regression filtering or not.
log.level	One of 'warn', 'error', 'info'
log.file	Log file to write to
var.threshold	Only consider genes and miRNA with variance > var.threshold. If this parameter is NULL no variance filtering is performed.
F.test	If true, an F-test is performed on each model parameter to assess its importance for the model based on the RSS of the full model vs the RSS of the nested model without the miRNA in question. This is time consuming and has the potential disadvantage that correlated miRNAs are removed even though they might play a role in ceRNA interactions. Use at your own risk.
F.test.p.adj.threshold	If F.test is TRUE, threshold to use for miRNAs to be included.
coefficient.threshold	threshold to cross for a regression coefficient to be called significant. depends on the parameter coefficient.direction.
coefficient.direction	If "<", coefficient has to be lower than coefficient.threshold, if ">", coefficient has to be larger than threshold. If NULL, the absolute value of the coefficient has to be larger than the threshold.

`select.non.targets`

For testing effect of miRNA target information. If TRUE, the method determines as usual which miRNAs are potentially targeting a gene. However, these are then replaced by a random sample of non-targeting miRNAs (without seeds) of the same size. Useful for testing if observed effects are caused by miRNA regulation.

`random_seed` A random seed to be used for reproducible results

`parallel.chunks`

Split into this number of tasks if parallel processing is set up. The number should be high enough to guarantee equal distribution of the work load in parallel execution. However, if the number is too large, e.g. in the worst case one chunk per computation, the overhead causes more computing time than can be saved by parallel execution. Register a parallel backend that is compatible with foreach to use this feature. More information can be found in the documentation of the foreach / doParallel packages.

## Value

A list of genes, where for each gene, the regulating miRNA are included as a data frame. For `F.test = TRUE` this is a data frame with `fstat` and `p-value` for each miRNA. Else it is a data frame with the model coefficients.

## See Also

`sponge`

## Examples

```
#library(doParallel)
#cl <- makePSOCKcluster(2)
#registerDoParallel(cl)
genes_miRNA_candidates <- sponge_gene_miRNA_interaction_filter(
  gene_expr = gene_expr,
  mir_expr = mir_expr,
  mir_predicted_targets = targetscan_symbol)
#stopCluster(cl)

#If we also perform an F-test, only few of the above miRNAs remain
genes_miRNA_candidates <- sponge_gene_miRNA_interaction_filter(
  gene_expr = gene_expr,
  mir_expr = mir_expr,
  mir_predicted_targets = targetscan_symbol,
  F.test = TRUE,
  F.test.p.adj.threshold = 0.05)
```

---

sponge_network	<i>Prepare a sponge network for plotting</i>
----------------	--

---

**Description**

Prepare a sponge network for plotting

**Usage**

```
sponge_network(  
  sponge_result,  
  mir_data,  
  target.genes = NULL,  
  show.sponge.interaction = TRUE,  
  show.mirnas = "none",  
  min.interactions = 3  
)
```

**Arguments**

sponge_result	ceRNA interactions as produced by the sponge method.
mir_data	miRNA interactions as produced by sponge_gene_miRNA_interaction_filter
target.genes	a character vector to select a subset of genes
show.sponge.interaction	whether to connect ceRNAs
show.mirnas	one of none, shared, all
min.interactions	minimum degree of a gene to be shown

**Value**

a list of nodes and edges

**Examples**

```
sponge_network(ceRNA_interactions, mir_interactions)
```

---

`sponge_node_centralities`*Computes various node centralities*

---

**Description**

Computes degree, eigenvector centrality and betweenness centrality for the ceRNA interaction network induced by the results of the SPONGE method

**Usage**

```
sponge_node_centralities(sponge_result, directed = FALSE)
```

**Arguments**

`sponge_result` output of the sponge method  
`directed` Whether to consider the input network as directed or not.

**Value**

data table or data frame with gene, degree, eigenvector and betweenness

**See Also**

`sponge`

**Examples**

```
sponge_node_centralities(ceRNA_interactions)
```

---

`sponge_plot_network` *Plot a sponge network*

---

**Description**

Plot a sponge network

**Usage**

```
sponge_plot_network(  
  sponge_result,  
  mir_data,  
  layout = "layout.fruchterman.reingold",  
  force.directed = FALSE,  
  ...  
)
```

**Arguments**

sponge_result	ceRNA interactions as produced by the sponge method.
mir_data	miRNA interactions as produced by sponge_gene_miRNA_interaction_filter
layout	one of the layout methods supported in the visNetwork package
force.directed	whether to produce a force directed network, gets slow for large networks
...	further params for sponge_network

**Value**

shows a plot

**Examples**

```
sponge_plot_network(ceRNA_interactions, mir_interactions)
```

---

```
sponge_plot_network_centralities
      plot node network centralities
```

---

**Description**

plot node network centralities

**Usage**

```
sponge_plot_network_centralities(
  network_centralities,
  measure = "all",
  x = "degree",
  top = 5,
  base_size = 18
)
```

**Arguments**

network_centralities	a result from sponge_node_centralities()
measure	one of 'all', 'degree', 'ev' or 'btw'
x	plot against another column in the data table, defaults to degree
top	label the top x samples in the plot
base_size	size of the text in the plot

**Value**

a plot



**Examples**

```
## Not run:  
network_centralities <- sponge_node_centralities(ceRNA_interactions)  
sponge_plot_network_centralities(network_centralities)  
## End(Not run)
```

---

sponge\_plot\_simulation\_results

*Plot simulation results for different null models*

---

**Description**

Plot simulation results for different null models

**Usage**

```
sponge_plot_simulation_results(null_model_data)
```

**Arguments**

null\_model\_data  
the output of sponge\_build\_null\_model

**Value**

a ggplot2 object

**Examples**

```
sponge_plot_simulation_results(precomputed_null_model)
```

---

sponge\_run\_benchmark *run sponge benchmark where various settings, i.e. with or without regression, single or pooled miRNAs, are compared.*

---

**Description**

run sponge benchmark where various settings, i.e. with or without regression, single or pooled miRNAs, are compared.

**Usage**

```
sponge_run_benchmark(  
  gene_expr,  
  mir_expr,  
  mir_predicted_targets,  
  number_of_samples = 100,  
  number_of_datasets = 100,  
  number_of_genes_to_test = c(25),  
  compute_significance = FALSE,  
  folder = NULL  
)
```

**Arguments**

gene_expr	A gene expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
mir_expr	A miRNA expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
mir_predicted_targets	(a list of) mir interaction sources such as targetscan, etc.
number_of_samples	number of samples in the null model
number_of_datasets	number of datasets to sample from the null model
number_of_genes_to_test	a vector of numbers of genes to be tested, e.g. c(250,500)
compute_significance	whether to compute p-values
folder	where the results should be saved, if NULL no output to disk

**Value**

a list (regression, no regression) of lists (single miRNA, pooled miRNAs) of benchmark results

**Examples**

```
sponge_run_benchmark(gene_expr = gene_expr, mir_expr = mir_expr,  
  mir_predicted_targets = targetscan_symbol,  
  number_of_genes_to_test = c(10), folder = NULL)
```

---

sponge\_subsampling      *Sponge subsampling*

---

## Description

Sponge subsampling

## Usage

```
sponge_subsampling(
  subsample.n = 100,
  subsample.repeats = 10,
  subsample.with.replacement = FALSE,
  subsample.plot = FALSE,
  gene_expr,
  mir_expr,
  ...
)
```

## Arguments

subsample.n	the number of samples to be drawn in each round
subsample.repeats	how often should the subsampling be done?
subsample.with.replacement	logical, should we allow samples to be used repeatedly
subsample.plot	logical, should the results be plotted as box plots
gene_expr	A gene expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
mir_expr	A miRNA expression matrix with samples in rows and features in columns. Alternatively an object of class ExpressionSet.
...	parameters passed on to the sponge function

## Value

a summary of the results with mean and standard deviations of the correlation and sensitive correlation.

## References

sponge

## Examples

```
sponge_subsampling(gene_expr = gene_expr,
  mir_expr = mir_expr, mir_interactions = mir_interactions,
  subsample.n = 10, subsample.repeats = 1)
```

---

targetscan_ensg	<i>targetscan predicted miRNA gene interactions</i>
-----------------	---

---

**Description**

targetscan predicted miRNA gene interactions

**Usage**

targetscan\_ensg

**Format**

A matrix gene ensembl ids vs miRNA family names.  $\geq 1$  if interaction is predicted, 0 otherwise

**Source**

[http://www.targetscan.org/vert\\_71/](http://www.targetscan.org/vert_71/)

---

targetscan_symbol	<i>targetscan predicted miRNA gene interactions</i>
-------------------	---

---

**Description**

targetscan predicted miRNA gene interactions

**Usage**

targetscan\_symbol

**Format**

A matrix gene symbols vs miRNA family names.  $\geq 1$  if interaction is predicted, 0 otherwise

**Source**

[http://www.targetscan.org/vert\\_71/](http://www.targetscan.org/vert_71/)

---

test\_cancer\_gene\_expr *example test expression data for spongEffects*

---

**Description**

example test expression data for spongEffects

**Usage**

test\_cancer\_gene\_expr

**Format**

a matrix with gene expression data

---

test\_cancer\_metadata *example test sample meta data for spongEffects*

---

**Description**

example test sample meta data for spongEffects

**Usage**

test\_cancer\_metadata

**Format**

a data frame with sample meta data, SUBTYPE must be inside your dataframe

---

test\_cancer\_mir\_expr *example test miRNA data for spongEffects*

---

**Description**

example test miRNA data for spongEffects

**Usage**

test\_cancer\_mir\_expr

**Format**

a matrix with miRNA expression data

---

train\_cancer\_gene\_expr *example training expression data for spongEffects*

---

**Description**

example training expression data for spongEffects

**Usage**

train\_cancer\_gene\_expr

**Format**

a matrix with gene expression data

---

train\_cancer\_metadata *example training sample meta data for spongEffects*

---

**Description**

example training sample meta data for spongEffects

**Usage**

train\_cancer\_metadata

**Format**

a data frame with sample meta data, SUBTYPE must be inside your dataframe

---

train\_cancer\_mir\_expr *example training miRNA data for spongEffects*

---

**Description**

example training miRNA data for spongEffects

**Usage**

train\_cancer\_mir\_expr

**Format**

a matrix with miRNA expression data

---

train\_ceRNA\_interactions

*example train ceRNA interactions for spongEffects*

---

**Description**

example train ceRNA interactions for spongEffects

**Usage**

train\_ceRNA\_interactions

**Format**

(obtained by SPONGE method)

---

train\_network\_centralities

*example train network centralities for spongEffects*

---

**Description**

example train network centralities for spongEffects

**Usage**

train\_network\_centralities

**Format**

(obtained by SPONGE method)

# Index

- \* **datasets**
  - ceRNA\_interactions, 5
  - ensembl.df, 8
  - gene\_expr, 17
  - mir\_expr, 19
  - mir\_interactions, 19
  - mircode\_ensg, 18
  - mircode\_symbol, 18
  - precomputed\_cov\_matrices, 25
  - precomputed\_null\_model, 26
  - targetscan\_ensg, 44
  - targetscan\_symbol, 44
  - test\_cancer\_gene\_expr, 45
  - test\_cancer\_metadata, 45
  - test\_cancer\_mir\_expr, 45
  - train\_cancer\_gene\_expr, 46
  - train\_cancer\_metadata, 46
  - train\_cancer\_mir\_expr, 46
  - train\_ceRNA\_interactions, 47
  - train\_network\_centralities, 47
- build\_classifier\_central\_genes, 3
- calibrate\_model, 4
- ceRNA\_interactions, 5
- check\_and\_convert\_expression\_data, 6
- define\_modules, 6
- enrichment\_modules, 7
- ensembl.df, 8
- filter\_ceRNA\_network, 8
- fn\_combined centrality, 9
- fn\_discretize\_spongeeffects, 10
- fn\_elasticnet, 10
- fn\_exact\_match\_summary, 11
- fn\_filter\_network, 11
- fn\_gene\_miRNA\_F\_test, 12
- fn\_get\_model\_coef, 12
- fn\_get\_rss, 13
- fn\_get\_semi\_random\_OE, 13
- fn\_get\_shared\_miRNAs, 14
- fn\_OE\_module, 14
- fn\_RF\_classifier, 15
- fn\_weighted\_degree, 16
- gene\_expr, 17
- genes\_pairwise\_combinations, 16
- get\_central\_modules, 17
- mir\_expr, 19
- mir\_interactions, 19
- mircode\_ensg, 18
- mircode\_symbol, 18
- plot\_accuracy\_sensitivity\_specificity, 19
- plot\_confusion\_matrices, 20
- plot\_density\_scores, 21
- plot\_heatmaps, 22
- plot\_involved\_miRNAs\_to\_modules, 23
- plot\_top\_modules, 25
- precomputed\_cov\_matrices, 25
- precomputed\_null\_model, 26
- prepare\_metabric\_for\_spongeEffects, 26
- prepare\_tcga\_for\_spongeEffects, 27
- Random\_spongeEffects, 28
- sample\_zero\_mscor\_cov, 29
- sample\_zero\_mscor\_data, 30
- sponge, 31
- sponge\_build\_null\_model, 33
- sponge\_compute\_p\_values, 34
- sponge\_edge\_centralities, 35
- sponge\_gene\_miRNA\_interaction\_filter, 35
- sponge\_network, 38
- sponge\_node\_centralities, 39
- sponge\_plot\_network, 39
- sponge\_plot\_network\_centralities, 40



sponge\_plot\_simulation\_results, [41](#)  
sponge\_run\_benchmark, [41](#)  
sponge\_subsampling, [43](#)

targetscan\_ensg, [44](#)  
targetscan\_symbol, [44](#)  
test\_cancer\_gene\_expr, [45](#)  
test\_cancer\_metadata, [45](#)  
test\_cancer\_mir\_expr, [45](#)  
train\_cancer\_gene\_expr, [46](#)  
train\_cancer\_metadata, [46](#)  
train\_cancer\_mir\_expr, [46](#)  
train\_ceRNA\_interactions, [47](#)  
train\_network\_centralities, [47](#)