

# Package ‘mslp’

May 20, 2024

**Type** Package

**Title** Predict synthetic lethal partners of tumour mutations

**Version** 1.6.0

**Description** An integrated pipeline to predict the potential synthetic lethality partners (SLPs) of tumour mutations, based on gene expression, mutation profiling and cell line genetic screens data. It has built-in support for data from cBioPortal. The primary SLPs correlating with mutations in WT and compensating for the loss of function of mutations are predicted by random forest based methods (GENIE3) and Rank Products, respectively. Genetic screens are employed to identify consensus SLPs leads to reduced cell viability when perturbed.

**License** GPL-3

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.2.3

**Depends** R (>= 4.2.0)

**Imports** data.table (>= 1.13.0), doRNG, fmsb, foreach, magrittr, org.Hs.eg.db, pROC, randomForest, RankProd, stats, utils

**Suggests** BiocStyle, doFuture, future, knitr, rmarkdown, roxygen2, tinytest

**VignetteBuilder** knitr

**biocViews** Pharmacogenetics, Pharmacogenomics

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

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** 23e640d

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-19

**Author** Chunxuan Shao [aut, cre]

**Maintainer** Chunxuan Shao <chunxuan@outlook.com>

## Contents

comp_slp . . . . .	2
cons_slp . . . . .	3
corr_slp . . . . .	4
est_im . . . . .	5
example_compSLP . . . . .	6
example_comp_mut . . . . .	7
example_corrSLP . . . . .	7
example_corr_mut . . . . .	7
example_expr . . . . .	8
example_z . . . . .	8
genie3 . . . . .	8
getlink . . . . .	10
merge_slp . . . . .	10
pp_tcga . . . . .	11
scr_slp . . . . .	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

comp_slp	<i>Identify SLPs via compensation</i>
----------	---------------------------------------

---

### Description

Identify SLPs compensating for the loss of function of mutations. The up-regulated SLPs are selected via the rank products algorithm, with option `calculateProduct = FALSE` for a robust results and capacity on large datasets.

### Usage

```
comp_slp(
  zscore_data,
  mut_data,
  mutgene = NULL,
  positive_perc = 0.5,
  p_thresh = 0.01,
  ...
)
```

### Arguments

zscore_data	a matrix (genes by patients) reflecting gene expression related to wide type samples. For example, generated from <code>pp_tcga</code> .
mut_data	a data.table with columns "patientid" and "mut_entrez".
mutgene	identify SLPs for sepecific muatation (gene symbols). If NULL (by default), the intersection genes between zscore_data and mut_data are used.

**positive\_perc** keep genes with positive zscore in at least positive\_perc \* number of mutation patients.  
**p\_thresh** pvalue threshold to filter out results.  
 ... additional parameters to [RankProducts](#).

### Value

A data.table with predicted SLPs.

**mut\_entrez** Entrez ids of mutations.

**mut\_symbol** Gene symbols of mutations.

**slp\_entrez** Entrez ids of SLPs.

**slp\_symbol** Gene symbols of SLPs.

**pvalue** p\_value from [RankProducts](#).

**fdr** "BH" adjusted pvalue via [p.adjust](#).

### Examples

```

#- Toy examples, see vignette for more.
#- Add the parallel backend.
require(future)
require(doFuture)
plan(multisession, workers = 2)
data("example_z")
data("example_comp_mut")
res <- comp_slp(example_z, example_comp_mut)
plan(sequential)

```

---

cons_slp	<i>Identify consensus SLPs</i>
----------	--------------------------------

---

### Description

Identify consensus SLPs based on Cohen's Kappa or hypergeometric test.

### Usage

```
cons_slp(screen_slp, tumour_slp)
```

### Arguments

**screen\_slp** screen hits data annotated with SLPs information, generated by [scr\\_slp](#).

**tumour\_slp** the merged SLPs data predicted by [corr\\_slp](#) and [comp\\_slp](#).

## Details

Consensus SLPs are enriched screen hits that are SLPs of same mutations in different cell lines. For each common mutation, the SLPs predicted from human tumour data are used as the total sets. We used either Cohen's Kappa coefficient on a confusion matrix, or Hypergeometric test, to test the significance of overlapping of screen hits.

## Value

A data.table.

**mut\_entrez** Entrez ids of mutations.

**mut\_symbol** Gene symbols of mutations.

**cons\_slp\_entrez** Entrez ids of consensus SLPs.

**cons\_slp\_symbol** Gene symbols of Consensus SLPs.

**cell\_1, cell\_2** From which pair of cell lines the consensus SLPs predicted.

**judgement** Judgement based on Cohen's Kappa.

**kappa\_value** Cohen's Kappa coefficient

**pvalue** pvalue for Cohen's Kappa coefficient.

**fdr** "BH" adjusted pvalue via [p.adjust](#).

## References

Landis JR, Koch GG (1977) The measurement of observer agreement for categorical data. *Biometrics*, 33: 159-174.

## Examples

```
#- See the examples in the vignette.
if (FALSE) k_res <- cons_slp(scr_res, merged_res)
```

---

corr\_slp

*Identify SLPs via correlation*

---

## Description

Identify SLPs of mutations based on co-expression. GENIE3 is employed to find genes highly correlated with mutations in wide type patients.

## Usage

```
corr_slp(
  expr_data,
  mut_data,
  mutgene = NULL,
  im_thresh = 0.001,
  topgene = 2000,
  ...
)
```

**Arguments**

expr_data	an expression matrix, genes by patients.
mut_data	a data.table with columns "patientid" and "mut_entrez".
mutgene	identify SLPs for sepecific muatation (gene symbols). If NULL (by default), the intersection genes between expr_data and mut_data are used.
im_thresh	minimum importance threshold.
topgene	top N genes above the im_thresh.
...	further parameters to <a href="#">genie3</a> .

**Value**

A data.table with predicted SLPs.

**mut\_entrez** Entrez ids of mutations.

**mut\_symbol** Gene symbols of mutations.

**slp\_entrez** Entrez ids of SLPs.

**slp\_symbol** Gene symbols of SLPs.

**fdr** "BH" adjusted pvalue via [p.adjust](#).

**im** The importance value returned by [genie3](#).

**Examples**

```
#- Toy examples, see vignette for more.
require(future)
require(doFuture)
plan(multisession, workers = 2)
data("example_expr")
data("example_corr_mut")
res <- corr_slp(example_expr, example_corr_mut)
plan(sequential)
```

---

 est\_im

*Estimate the importance threshold for GENIE3*


---

**Description**

Estimate the importance threshold based on repetition GENIE3 results via ROC.

**Usage**

```
est_im(permu_data, fdr_thresh = 0.001)
```

**Arguments**

permu_data	permuated <a href="#">corr_slp</a> results.
fdr_thresh	fdr threshold to selected "TRUE" SLPs.

**Details**

We first generate a SLPs by repetition matrix from repetition GENIE3 results. SLPs with high im value in repetitions are selected and considered as "TRUE" SLPs via the rank product algorithm. Then for each repetition, we perform receiver operating characteristic curve analysis and select an optimal threshold by "youden" approach. The optimal thresholds are averaged to get the final threshold.

**Value**

A data.table with mut\_entrez (mutation entrez\_id) and roc\_thresh (estimated im threshold).

**Examples**

```
#- Toy examples.
require(future)
require(doFuture)
plan(multisession, workers = 2)
data(example_expr)
data(example_corr_mut)
mutgene <- sample(intersect(example_corr_mut$mut_entrez, rownames(example_expr)), 2)
nperm <- 5
res <- lapply(seq_len(nperm), function(x) corr_slp(example_expr,
  example_corr_mut, mutgene = mutgene))
roc_thresh <- est_im(res)
plan(sequential)
```

---

example\_compSLP

*SLPs predicted by comp\_slp*

---

**Description**

SLPs predicted by comp\_slp

**Usage**

```
data(example_compSLP)
```

**Format**

A data.table.

---

example_comp_mut	<i>Patients mutations to be use in the comp_slp</i>
------------------	---

---

**Description**

Mutations and related TCGA ids.

**Usage**

```
data(example_comp_mut)
```

**Format**

A data.table.

---

example_corrSLP	<i>SLPs predicted by corr_slp</i>
-----------------	-----------------------------------

---

**Description**

SLPs predicted by corr\_slp

**Usage**

```
data(example_corrSLP)
```

**Format**

A data.table.

---

example_corr_mut	<i>Patients mutations to be use in the corr_slp</i>
------------------	---

---

**Description**

Mutations and related TCGA ids.

**Usage**

```
data(example_corr_mut)
```

**Format**

A data.table.

---

example_expr	<i>Expression data to be used in comp_slp</i>
--------------	---

---

**Description**

Expresion matrix, genes by samples.

**Usage**

```
data(example_expr)
```

**Format**

A matrix.

---

example_z	<i>Expression data to be used in corr_slp</i>
-----------	---

---

**Description**

Z score matrix, genes by samples.

**Usage**

```
data(example_z)
```

**Format**

A matrix.

---

genie3	<i>Run GENIE3</i>
--------	-------------------

---

**Description**

Calculate the weight matrix between genes via randomForest, modified from original codes by Huynh-Thu, V.A.



**Usage**

```
genie3(
  expr.matrix,
  ngene = NULL,
  K = "sqrt",
  nb.trees = 1000,
  input.idx = NULL,
  importance.measure = "IncNodePurity",
  trace = FALSE,
  ...
)
```

**Arguments**

<code>expr.matrix</code>	expression matrix (genes by samples).
<code>ngene</code>	an integer, only up to the first <code>ngene</code> (included) targets (responsible variables).
<code>K</code>	choice of number of input genes randomly, must be one of "sqrt", "all", an integer.
<code>nb.trees</code>	number of trees in ensemble for each target gene (default 1000).
<code>input.idx</code>	subset of genes used as input genes (default all genes). A vector of indices or gene names is accepted.
<code>importance.measure</code>	type of variable importance measure, "IncNodePurity" or "%IncMSE".
<code>trace</code>	index of currently computed gene is reported (default FALSE).
<code>...</code>	parameter to <code>randomForest</code> .

**Value**

A weighted adjacency matrix of inferred network, element  $w_{ij}$  (row  $i$ , column  $j$ ) gives the importance of the link from regulatory gene  $i$  to target gene  $j$ .

**References**

Huynh-Thu, V.A., Irrthum, A., Wehenkel, L., and Geurts, P. (2010). Inferring Regulatory Networks from Expression Data Using Tree-Based Methods. *PLoS ONE* 5, e12776.

**Examples**

```
#- Toy examples.
mtx <- matrix(sample(1000, 100), nrow = 5)
mtx <- rbind(mtx[1, ] * 2 + rnorm(20), mtx)
colnames(mtx) <- paste0("s_", seq_len(ncol(mtx)))
rownames(mtx) <- paste0("g_", seq_len(nrow(mtx)))
res <- genie3(mtx, nb.trees = 100)
```

---

getlink	<i>Get sorted list of regulatory links in GENIE3 results</i>
---------	--

---

**Description**

Take genie3 output and sort the links.

**Usage**

```
getlink(weight.matrix, report.max = NULL)
```

**Arguments**

`weight.matrix` a weighted adjacency matrix as returned by `genie3`.  
`report.max` maximum number of links to report (default all links).

**Value**

A `data.table` of links with columns "from.gene", "to.gene", "im".

**Examples**

```
mtx <- matrix(sample(1000, 100), nrow = 5)
mtx <- rbind(mtx[1, ] * 2 + rnorm(20), mtx)
colnames(mtx) <- paste0("s_", seq_len(ncol(mtx)))
rownames(mtx) <- paste0("g_", seq_len(nrow(mtx)))
res <- genie3(mtx, nb.trees = 10)
res_link <- getlink(res)
```

---

merge_slp	<i>Merge SLPs</i>
-----------	-------------------

---

**Description**

Merge predicted SLPs from `comp_slp` and `corr_slp`.

**Usage**

```
merge_slp(comp_data, corr_data)
```

**Arguments**

`comp_data` predicted SLPs from `comp_slp`.  
`corr_data` predicted SLPs from `corr_slp`.

**Value**

A data.table.

**mut\_entrez** Entrez ids of mutations.

**mut\_symbol** Gene symbols of mutations.

**slp\_entrez** Entrez ids of SLPs.

**slp\_symbol** Gene symbols of SLPs.

**pvalue** p\_value from [RankProducts](#).

**fdr** "BH" adjusted pvalue via [p.adjust](#).

**im** The importance value returned by [genie3](#).

**dualhit** Whether the slp is identified by [corr\\_slp](#) and [comp\\_slp](#).

**Examples**

```
data("example_z")
data("example_comp_mut")
comp_res <- comp_slp(example_z, example_comp_mut)

data("example_expr")
data("example_corr_mut")
corr_res <- corr_slp(example_expr, example_corr_mut)

res <- merge_slp(comp_res, corr_res)
```

---

pp\_tcga

*Process tumour genomic data*

---

**Description**

Preprocess mutation, cna, expression and zscore datasets in TCGA PanCancer Atlas by cBioPortal.

**Usage**

```
pp_tcga(  
  p_mut,  
  p_cna,  
  p_exprs,  
  p_score,  
  freq_thresh = 0.02,  
  expr_thresh = 10,  
  hypermut_thresh = 300  
)
```

**Arguments**

p_mut	path of mutation data, like "data_mutations_uniprot.txt" provided by cBioPortal.
p_cna	path of copy number variation data, like "data_CNA.txt".
p_exprs	path of normalized RNAseq expression data, like "data_RNA_Seq_v2_expression_median.txt".
p_score	path of zscore data, like "data_RNA_Seq_v2_mRNA_median_Zscores.txt".
freq_thresh	threshold to select recurrent mutations.
expr_thresh	threshold to remove low expression genes.
hypermut_thresh	threshold for hypermutations.

**Details**

It is designed to process the TCGA data provided by cBioPortal. In mutation data, "Missense\_Mutation", "Nonsense\_Mutation", "Frame\_Shift\_Del", "Frame\_Shift\_Ins", "In\_Frame\_Del", "In\_Frame\_Ins", "Nonstop\_Mutation" are selected for the downstream analysis. In CNA data, genes with GISTIC value equal to -2 are used. Patients with hypermutations are removed. Low expression genes, or genes that are not detected in any patient are filtered out.

**Value**

Return a list of mut\_data, expr\_data and zscore\_data, while expr\_data and zscore\_data are matrix (entrez\_id by patients), mut\_data is a data.table with two columns of "patientid" and "mut\_entrez".

**References**

Cerami et al. The cBio Cancer Genomics Portal: An Open Platform for Exploring Multidimensional Cancer Genomics Data. *Cancer Discovery*. May 2012 2; 401. Gao et al. Integrative analysis of complex cancer genomics and clinical profiles using the cBioPortal. *Sci. Signal*. 6, p11 (2013).

**Examples**

```
#- See vignette for more details.
if (FALSE) {
  P_mut <- "data_mutations_extended.txt"
  P_cna <- "data_CNA.txt"
  P_expr <- "data_RNA_Seq_v2_expression_median.txt"
  P_z <- "data_RNA_Seq_v2_mRNA_median_Zscores.txt"
  res <- pp_tcg(P_mut, P_cna, P_expr, P_z)
  saveRDS(res$mut_data, "mut_data.rds")
  saveRDS(res$expr_data, "expr_data.rds")
  saveRDS(res$zscore_data, "zscore_data.rds")
}
```

---

scr_slp	<i>Identify SLPs in screen hits</i>
---------	-------------------------------------

---

### Description

Identify whether screen hits are SLPs of mutations detected in both patients and cell lines, based on predicted SLPs in [corr\\_slp](#) and [comp\\_slp](#).

### Usage

```
scr_slp(cell, screen_data, cell_mut, tumour_slp)
```

### Arguments

<code>cell</code>	a cell line.
<code>screen_data</code>	a data.table of genomic screen results with three columns, "screen_entrez", "screen_symbol" and "cell_line".
<code>cell_mut</code>	cell line mutation data.
<code>tumour_slp</code>	merged SLPs.

### Value

A data.table.

**cell\_line** Name of cell lines.

**screen\_entrez** Entrez ids of hits.

**screen\_symbol** Gene symbols of hits.

**mut\_entrez** Entrez ids of mutations.

**mut\_symbol** Gene symbols of mutations.

**is\_slp** Whether the targeted gene is a SLP.

**pvalue** p\_value from [RankProducts](#).

**fdr** "BH" adjusted pvalue via [p.adjust](#).

**im** The importance value returned by [genie3](#).

**dualhit** Whether the slp is identified by [corr\\_slp](#) and [comp\\_slp](#).

### Examples

```
require(future)
require(doFuture)
plan(multisession, workers = 2)
library(magrittr)
library(data.table)
data(example_compSLP)
data(example_corrSLP)
merged_res <- merge_slp(example_compSLP, example_corrSLP)
```

```
##- Toy hits data.
screen_1 <- merged_res[, .(slp_entrez, slp_symbol)] %>%
  unique %>%
  .[sample(nrow(.), round(.8 * nrow(.)))] %>%
  setnames(c(1, 2), c("screen_entrez", "screen_symbol")) %>%
  .[, cell_line := "cell_1"]

screen_2 <- merged_res[, .(slp_entrez, slp_symbol)] %>%
  unique %>%
  .[sample(nrow(.), round(.8 * nrow(.)))] %>%
  setnames(c(1, 2), c("screen_entrez", "screen_symbol")) %>%
  .[, cell_line := "cell_2"]

screen_hit <- rbind(screen_1, screen_2)

##- Toy mutations data.
mut_1 <- merged_res[, .(mut_entrez)] %>%
  unique %>%
  .[sample(nrow(.), round(.8 * nrow(.)))] %>%
  .[, cell_line := "cell_1"]

mut_2 <- merged_res[, .(mut_entrez)] %>%
  unique %>%
  .[sample(nrow(.), round(.8 * nrow(.)))] %>%
  .[, cell_line := "cell_2"]

mut_info <- rbind(mut_1, mut_2)

##- Hits that are identified as SLPs.
scr_res <- lapply(c("cell_1", "cell_2"), scr_slp, screen_hit, mut_info, merged_res)
scr_res[lengths(scr_res) == 0] <- NULL
scr_res <- rbindlist(scr_res)
plan(sequential)
```

# Index

## \* datasets

- example\_comp\_mut, 7
- example\_compSLP, 6
- example\_corr\_mut, 7
- example\_corrSLP, 7
- example\_expr, 8
- example\_z, 8

comp\_slp, 2, 3, 10, 11, 13  
cons\_slp, 3  
corr\_slp, 3, 4, 5, 10, 11, 13

est\_im, 5  
example\_comp\_mut, 7  
example\_compSLP, 6  
example\_corr\_mut, 7  
example\_corrSLP, 7  
example\_expr, 8  
example\_z, 8

genie3, 5, 8, 11, 13  
getlink, 10

merge\_slp, 10

p.adjust, 3–5, 11, 13  
pp\_tcga, 2, 11

RankProducts, 3, 11, 13

scr\_slp, 3, 13