# Package ‘scregclust’

July 23, 2025

**Title** Reconstructing the Regulatory Programs of Target Genes in scRNA-Seq Data

**Version** 0.2.0

**Description** Implementation of the scregclust algorithm described in Larsson, Held, et al. (2024) <doi:10.1038/s41467-024-53954-3> which reconstructs regulatory programs of target genes in scRNA-seq data. Target genes are clustered into modules and each module is associated with a linear model describing the regulatory program.

**Encoding** UTF-8

**Depends** R (>= 3.5.0)

**Imports** Matrix, stats, methods, utils, reshape, igraph, graphics, grid, cli, prettyunits, ggplot2, rlang, Rcpp (>= 1.0.8)

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), Seurat (>= 5.0.0), glmGamPoi, hdf5r

**LinkingTo** Rcpp, RcppEigen

**VignetteBuilder** knitr

**RoxygenNote** 7.3.2

**License** GPL (>= 3)

**Config/testthat/edition** 3

**URL** `https://scmethods.github.io/scregclust/`, `https://github.com/scmethods/scregclust/`

**BugReports** `https://github.com/scmethods/scregclust/issues`

**NeedsCompilation** yes

**Author** Felix Held [aut, cre] (ORCID: <`https://orcid.org/0000-0002-7679-7752`>), Ida Larsson [aut] (ORCID: <`https://orcid.org/0000-0001-5422-4243`>), Sven Nelander [aut] (ORCID: <`https://orcid.org/0000-0003-1758-1262`>), André Armatowski [ctb]

**Maintainer** Felix Held <`felix.held@gmail.com`>

**Repository** CRAN

**Date/Publication** 2024-12-06 14:30:01 UTC

# Contents

---

`available_results` *Extract final configurations into a data frame*

---

## Description

Extract final configurations into a data frame

## Usage

```
available_results(obj)
```

## Arguments

| | |
|---|---|
| `obj` | An object of class `scregclust` |

## Value

A `data.frame` containing penalization parameters and final configurations for those penalizations.

## `cluster_overlap` *Create a table of module overlap for two clusterings*

### Description

Compares two clusterings and creates a table of overlap between them. Module labels do not have to match.

### Usage

```
cluster_overlap(k1, k2)
```

### Arguments

| | |
|---|---|
| `k1` | First clustering |
| `k2` | Second clustering |

### Value

A matrix showing the module overlap with the labels of `k1` in the columns and the labels of `k2` in the rows.

## `fast_cor` *Fast computation of correlation*

### Description

This uses a more memory-intensive but much faster algorithm than the built-in `cor` function.

### Usage

```
fast_cor(x, y)
```

### Arguments

| | |
|---|---|
| `x` | first input matrix |
| `y` | second input matrix |

### Details

Computes the correlation between the columns of `x` and `y`.

### Value

Correlations matrix between the columns of `x` and `y`

## `find_module_sizes` *Determine module sizes*

### Description

Determine module sizes

### Usage

```
find_module_sizes(module, n_modules)
```

### Arguments

| | |
|---|---|
| `module` | Vector of module indices |
| `n_modules` | Total number of modules |

### Value

A named vector containing the name of the module (its index or `"Noise"`) and the number of elements in that module

## `get_avg_num_regulators` *Get the average number of active regulators per module*

### Description

Get the average number of active regulators per module

### Usage

```
get_avg_num_regulators(fit)
```

### Arguments

| | |
|---|---|
| `fit` | An object of class `scRegClust` |

### Value

A `data.frame` containing the average number of active regulators per module for each penalization parameter.

## `get_num_final_configs` *Return the number of final configurations*

### Description

Returns the number of final configurations per penalization parameter in an scRegClust object.

### Usage

```
get_num_final_configs(fit)
```

### Arguments

| | |
|---|---|
| `fit` | An object of class `scRegClust` |

### Value

An integer vector containing the number of final configurations for each penalization parameter.

## `get_rand_indices` *Compute Rand indices*

### Description

Compute Rand indices for fitted scregclust object

### Usage

```
get_rand_indices(fit, groundtruth, adjusted = TRUE)
```

### Arguments

| | |
|---|---|
| `fit` | An object of class `scregclust` |
| `groundtruth` | A known clustering of the target genes (integer vector) |
| `adjusted` | If TRUE, the Adjusted Rand index is computed. Otherwise the ordinary Rand index is computed. |

### Value

A `data.frame` containing the Rand indices. Since there can be more than one final configuration for some penalization parameters, Rand indices are averaged for each fixed penalization parameter. Returned are the mean, standard deviation and number of final configurations that were averaged.

### References

W. M. Rand (1971). "Objective criteria for the evaluation of clustering methods". Journal of the American Statistical Association 66 (336): 846–850. DOI:10.2307/2284239

Lawrence Hubert and Phipps Arabie (1985). "Comparing partitions". Journal of Classification. 2 (1): 193–218. DOI:10.1007/BF01908075

---

## `get_regulator_list` *Return list of regulator genes*

---

### Description

Return list of regulator genes

### Usage

```
get_regulator_list(mode = c("TF", "kinase"))
```

### Arguments

| | |
|---|---|
| `mode` | Determines which genes are considered to be regulators. Currently supports TF=transcription factors and kinases. |

### Value

a list of gene symbols

### See Also

`scregclust_format()`

---

## `get_target_gene_modules` *Extract target gene modules for given penalization parameters*

---

### Description

Extract target gene modules for given penalization parameters

### Usage

```
get_target_gene_modules(fit, penalization = NULL)
```

### Arguments

| | |
|---|---|
| `fit` | An object of class `scregclust` |
| `penalization` | A numeric vector of penalization parameters. The penalization parameters specified here must have been used used during fitting of the `fit` object. |

### Value

A list of lists of final target modules. One list for each parameter in `penalization`. The lists contain the modules of target genes for each final configuration.

---

## `kmeanspp` *Perform the k-means++ algorithm*

---

### Description

Performs the k-means++ algorithm to cluster the rows of the input matrix.

### Usage

```
kmeanspp(x, n_cluster, n_init_clusterings = 10L, n_max_iter = 10L)
```

### Arguments

| | |
|---|---|
| `x` | Input matrix (n x p) |
| `n_cluster` | Number of clusters |
| `n_init_clusterings` | Number of repeated random initializations to perform |
| `n_max_iter` | Number of maximum iterations to perform in the k-means algorithm |

### Details

Estimation is repeated

### Value

An object of class `stats::kmeans`.

### References

David Arthur and Sergei Vassilvitskii. K-Means++: The advantages of careful seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA ’07, pages 1027—1035. Society for Industrial and Applied Mathematics, 2007.

## `plot_module_count_helper` *Plot average silhouette scores and average predictive $R\hat{}2$*

### Description

Plot average silhouette scores and average predictive $R^2$

### Usage

```
plot_module_count_helper(list_of_fits, penalization)
```

### Arguments

| | |
|---|---|
| `list_of_fits` | A list of `scregclust` objects each fit to the same dataset across a variety of module counts (varying `n_modules` while running `scregclust`). |
| `penalization` | Either a single numeric value requesting the results for the same penalty parameter across all fits in `list_of_fits`, or one for each individual fit. |

### Value

A ggplot2 plot showing the average silhouette score and the average predictive $R^2$

## `plot_regulator_network` *Plotting the regulatory table from scregclust as a directed graph*

### Description

Plotting the regulatory table from scregclust as a directed graph

### Usage

```
plot_regulator_network(
  output,
  arrow_size = 0.3,
  edge_scaling = 30,
  no_links = 6,
 col = c("gray80", "#FC7165", "#BD828C", "#9D8A9F", "#7D92B2", "#BDA88C", "#FCBD65",
   "#F2BB90", "#E7B9BA", "#BDB69C", "#92B27D", "#9B8BA5", "#9D7DB2", "#94A5BF")
)
```

### Arguments

| | |
|---|---|
| `output` | Object of type `scregclust_output` from a fit of the scregclust algorithm. |
| `arrow_size` | Size of arrow head |
| `edge_scaling` | Scaling factor for edge width |
| `no_links` | Threshold value (0-10) for number of edges to show, higher value = more stringent threshold = less edges |
| `col` | color |

### Value

Graph with gene modules and regulators as nodes

---

## `plot_silhouettes` *Plot individual silhouette scores*

---

### Description

Plot individual silhouette scores

### Usage

```
plot_silhouettes(list_of_fits, penalization, final_config = 1L)
```

### Arguments

| | |
|---|---|
| `list_of_fits` | A list of `scregclust` objects each fit to the same dataset across a variety of module counts (varying `n_modules` when running `scregclust`). |
| `penalization` | Either a single numeric value requesting the results for the same penalty parameter across all fits in `list_of_fits`, or one for each individual fit. |
| `final_config` | The final configuration that should be visualized. Either a single number to be used for all fits in `list_of_fits`, or one for each individual fit. |

### Value

A ggplot2 plot showing the the silhouette scores for each supplied fit.

`scregclust` *Uncover gene modules and their regulatory programs from single-cell data*

## Description

Use the scRegClust algorithm to determine gene modules and their regulatory programs from single-cell data.

## Usage

```
scregclust(
  expression,
  genesymbols,
  is_regulator,
  penalization,
  n_modules,
  initial_target_modules = NULL,
  sample_assignment = NULL,
  center = TRUE,
  split1_proportion = 0.5,
  total_proportion = 1,
  split_indices = NULL,
  prior_indicator = NULL,
  prior_genesymbols = NULL,
  prior_baseline = 1e-06,
  prior_weight = 0.5,
  min_module_size = 0L,
  allocate_per_obs = TRUE,
  noise_threshold = 0.025,
  n_cycles = 50L,
  use_kmeanspp_init = TRUE,
  n_initializations = 50L,
  max_optim_iter = 10000L,
  tol_coop_rel = 1e-08,
  tol_coop_abs = 1e-12,
  tol_nnls = 1e-04,
  compute_predictive_r2 = TRUE,
  compute_silhouette = FALSE,
  nowarnings = FALSE,
  verbose = TRUE,
  quick_mode = FALSE,
  quick_mode_percent = 0.1
)
```

## Arguments

| | |
|---|---|
| `expression` | `p x n` matrix of pre-processed single cell expression data with `p` rows of genes and `n` columns of cells. |
| `genesymbols` | A vector of gene names corresponding to rows of `expression`. Has to be of length `p`. |
| `is_regulator` | An indicator vector where `1` indicates that the corresponding row in `expression` is a candidate regulator. All other rows represent target genes. Has to be of length `p`. |
| `penalization` | Sparsity penalty related to the amount of regulators associated with each module. Either a single positive number or a vector of positive numbers. |
| `n_modules` | Requested number of modules (integer). If this is provided without specifying `initial_target_modules`, then an initial module allocation is performed on the cross-correlation matrix of targets and genes on the first dataset after data splitting. |
| `initial_target_modules` | The initial assignment of target genes to modules of length `sum(is_regulator == 0L)`. If this is not specified, then see `n_modules` regarding module initialization. If provided, `use_kmeanspp_init` and `n_initializations` are ignored. |
| `sample_assignment` | A vector of sample assignment for each cell, can be used to perform the data splitting with stratification. Has to be of length `n`. No stratification if `NULL` is supplied. |
| `center` | Whether or not genes should be centered within each subgroup defined in `sample_assignment`. |
| `split1_proportion` | The proportion to use for the first dataset during data splitting. The proportion for the second dataset is `1 - split1_proportion`. If stratification with `sample_assignment` is used, then the proportion of each strata is controlled. |
| `total_proportion` | Can be used to only use a proportion of the supplied observations. The proportion of the first dataset during data splitting in relation to the full dataset will be `total_proportion * split1_proportion`. |
| `split_indices` | Can be used to provide an explicit data split. If this is supplied then `split1_proportion`, and `total_proportion` are ignored. Note that if `sample_assigment` is provided and `center == TRUE`, then subgroup centering will be performed as in the case of random splitting. A vector of length `n` containing entries 1 for cells in the first data split, 2 for cells in the second data split and `NA` for cells that should be excluded from the computations. |
| `prior_indicator` | An indicator matrix (sparse or dense) of size `q x q` that indicates whether there is a known functional relationship between two genes. Ideally, this is supplied as a sparse matrix (`sparseMatrix` in the `Matrix` package). If not, then the matrix is converted to one. |
| `prior_genesymbols` | A vector of gene names of length q corresponding to the rows/columns in `prior_indicator`. Does not have to be the same as `genesymbols`, but only useful if there is overlap. |

| | |
|---|---|
| `prior_baseline` | A positive baseline for the network prior. The larger this parameter is, the less impact the network prior will have. |
| `prior_weight` | A number between 0 and 1 indicating the strength of the prior in relation to the data. 0 ignores the prior and makes the algorithm completely data-driven. 1 uses only the prior during module allocation. |
| `min_module_size` | Minimum required size of target genes in a module. Smaller modules are emptied. |
| `allocate_per_obs` | Whether module allocation should be performed for each observation in the second data split separately. If `FALSE`, target genes are allocated into modules on the aggregate sum of squares across all observations in the second data split. |
| `noise_threshold` | Threshold for the best $R^2$ of a target gene before it gets identified as noise. |
| `n_cycles` | Number of maximum algorithmic cycles. |
| `use_kmeanspp_init` | Use kmeans++ for module initialization if `initial_target_modules` is a single integer; otherwise use kmeans with random initial cluster centers |
| `n_initializations` | Number of kmeans(++) initialization runs. |
| `max_optim_iter` | Maximum number of iterations during optimization in the coop-Lasso and NNLS steps. |
| `tol_coop_rel` | Relative convergence tolerance during optimization in the coop-Lasso step. |
| `tol_coop_abs` | Absolute convergence tolerance during optimization in the coop-Lasso step. |
| `tol_nnls` | Convergence tolerance during optimization in the NNLS step. |
| `compute_predictive_r2` | Whether to compute predictive $R^2$ per module as well as regulator importance. |
| `compute_silhouette` | Whether to compute silhouette scores for each target gene. |
| `nowarnings` | When turned on then no warning messages are shown. |
| `verbose` | Whether to print progress. |
| `quick_mode` | Whether to use a reduced number of noise targets to speed up computations. |
| `quick_mode_percent` | A number in [0, 1) indicating the amount of noise targets to use in the re-allocation process if `quick_mode = TRUE`. |

## Value

A list with S3 class `scregclust` containing

| | |
|---|---|
| `penalization` | The supplied `penalization` parameters |
| `results` | A list of result lists (each with S3 class `scregclust_result`), one for each supplied `penalization` parameter. See below. |
| `initial_target_modules` | Initial allocation of target genes into modules. |

split_indices either verbatim the vector given as input or a vector encoding the splits as NA = not included, 1 = split 1 or 2 = split 2. Allows reproducibility of data splits.

For each supplied penalization parameter, results contains a list with

- the current penalization parameter,
- the supplied genesymbols after filtering (as used during fitting),
- the supplied is_regulator vector after filtering (as used during fitting),
- the number of fitted modules n_modules,
- whether the current run converged to a single configuration (as a boolean),
- as well as an output object containing the numeric results for each final configuration.

It is possible that the algorithm ends in a finite cycle of configurations instead of a unique final configuration. Therefore, output is a list with each element itself being a list with the following contents:

reg_table a regulator table, a matrix of weights for each regulator and module

module vector of same length as genesymbols containing the module assignments for all genes with regulators marked as NA. Genes considered noise are marked as -1.

module_all same as module, however, genes that were marked as noise (-1 in module) are assigned to the module in which it has the largest $R^2$, even if it is below noise_threshold.

r2 matrix of predictive $R^2$ value for each target gene and module

best_r2 vector of best predictive $R^2$ for each gene (regulators marked with NA)

best_r2_idx module index corresponding to best predictive $R^2$ for each gene (regulators marked with NA)

r2_module a vector of predictive $R^2$ values for each module (included if compute_predictive_r2 == TRUE)

importance a matrix of importance values for each regulator (rows) and module (columns) (included if compute_predictive_r2 == TRUE)

r2_cross_module_per_target a matrix of cross module $R^2$ values for each target gene (rows) and each module (columns) (included if compute_silhouette == TRUE)

silhouette a vector of silhouette scores for each target gene (included if compute_silhouette == TRUE)

models regulator selection for each module as a matrix with regulators in rows and modules in columns

signs regulator signs for each module as a matrix with regulators in rows and modules in columns

weights average regulator coefficient for each module

coeffs list of regulator coefficient matrices for each module for all target genes as re-estimated in the NNLS step

sigmas matrix of residual variances, one per target gene in each module; derived from the residuals in NNLS step

## `scregclust_format` *Package data before clustering*

### Description

Package data before clustering

### Usage

```
scregclust_format(expression_matrix, mode = c("TF", "kinase"))
```

### Arguments

| | |
|---|---|
| `expression_matrix` | The p x n gene expression matrix with gene symbols as rownames. |
| `mode` | Determines which genes are considered to be regulators. |

### Value

A list with

| | |
|---|---|
| `genesymbols` | The gene symbols extracted from the expression matrix |
| `sample_assignment` | A vector filled with 1's of the same length as there are columns in the gene expression matrix. |
| `is_regulator` | Whether a gene is considered to be a regulator or not, determined dependent on `mode`. |

### See Also

`get_regulator_list()`

# Index