Package: coglasso (via r-universe)

September 8, 2024

Type Package

Title Collaborative Graphical Lasso - Multi-Omics Network Reconstruction

Version 1.0.2.9000

Description Reconstruct networks from multi-omics data sets with the collaborative graphical lasso (coglasso) algorithm described in Albanese, A., Kohlen, W., and Behrouzi, P. (2024) <arXiv:2403.18602>. Build multiple networks using the coglasso() function, select the best one with stars_coglasso().

URL https://github.com/DrQuestion/coglasso, https://drquestion.github.io/coglasso/

BugReports https://github.com/DrQuestion/coglasso/issues

License GPL (>= 2)

Imports igraph, lifecycle, Matrix, Rcpp (>= 1.0.11), rlang, stats, utils

LinkingTo Rcpp, RcppEigen

Depends R (>= 2.10)

LazyData true

Encoding UTF-8

RoxygenNote 7.2.3

Roxygen list(markdown = TRUE)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Repository https://drquestion.r-universe.dev

RemoteUrl https://github.com/drquestion/coglasso

RemoteRef HEAD

RemoteSha 6c8ff6e65ead675c9ecff7e4b7ca2970579af544

Contents

bs 2
coglasso 6
get_network 9
multi_omics_sd 10
plot.select_coglasso 11
select_coglasso 12
xestars 15
xstars 18

Index **21**

`bs` *Build multiple networks and select the best one from a multi-omics data set*

Description

`bs()` wraps the two main functions of the package in a single one: `coglasso()`, to build multiple multi-omics networks, and `select_coglasso()` to select the best one according to the chosen criterion.

Usage

```
bs(
  data,
  p = NULL,
  pX = lifecycle::deprecated(),
  lambda_w = NULL,
  lambda_b = NULL,
  c = NULL,
  nlambda_w = NULL,
  nlambda_b = NULL,
  nc = NULL,
  lambda_w_max = NULL,
  lambda_b_max = NULL,
  c_max = NULL,
  lambda_w_min_ratio = NULL,
  lambda_b_min_ratio = NULL,
  c_min_ratio = NULL,
  cov_output = FALSE,
  method = "xestars",
  stars_thresh = 0.1,
  stars_subsample_ratio = NULL,
  rep_num = 20,
  max_iter = 10,
  old_sampling = FALSE,
```

```
  light = TRUE,
  ebic_gamma = 0.5,
  verbose = TRUE
)
```

Arguments

data	The input multi-omics data set. Rows should be samples, columns should be variables. Variables should be grouped by their assay (e.g. transcripts first, then metabolites). data is a required parameter.
p	A vector with with the number of variables for each omic layer of the data set (e.g. the number of transcripts, metabolites etc.), in the same order the layers have in the data set. If given a single number, coglasso() assumes that the total of data sets is two, and that the number given is the dimension of the first one.
pX	**[Deprecated]** pX is no longer supported. Please use p.
lambda_w	A vector of values for the parameter λ_w, the penalization parameter for the "within" interactions. Overrides nlambda_w.
lambda_b	A vector of values for the parameter λ_b, the penalization parameter for the "between" interactions. Overrides nlambda_b.
c	A vector of values for the parameter c, the weight given to collaboration. Overrides nc.
nlambda_w	The number of requested λ_w parameters to explore. A sequence of size nlambda_w of λ_w parameters will be generated. Defaults to 8. Ignored when lambda_w is set by the user.
nlambda_b	The number of requested λ_b parameters to explore. A sequence of size nlambda_b of λ_b parameters will be generated. Defaults to 8. Ignored when lambda_b is set by the user.
nc	The number of requested c parameters to explore. A sequence of size nc of c parameters will be generated. Defaults to 8. Ignored when c is set by the user.
lambda_w_max	The greatest generated λ_w. By default it is computed with a data-driven approach. Ignored when lambda_w is set by the user.
lambda_b_max	The greatest generated λ_b. By default it is computed with a data-driven approach. Ignored when lambda_b is set by the user.
c_max	The greatest generated c. Defaults to 10. Ignored when c is set by the user.
lambda_w_min_ratio	The ratio of the smallest generated λ_w over the greatest generated λ_w. Defaults to 0.1. Ignored when lambda_w is set by the user.
lambda_b_min_ratio	The ratio of the smallest generated λ_b over the greatest generated λ_b. Defaults to 0.1. Ignored when lambda_b is set by the user.
c_min_ratio	The ratio of the smallest generated c over the greatest generated c. Defaults to 0.1. Ignored when c is set by the user.
cov_output	Add the estimated variance-covariance matrix to the output.
method	The model selection method to select the best combination of hyperparameters. The available options are "xstars", "xestars" and "eBIC". Defaults to "xestars".

`stars_thresh` The threshold set for variability of the explored networks at each iteration of the algorithm. The λ_w or the λ_b associated to the most stable network before the threshold is overcome is selected.

`stars_subsample_ratio` The proportion of samples in the multi-omics data set to be randomly subsampled to estimate the variability of the network under the given hyperparameters setting. Defaults to 80% when the number of samples is smaller than 144, otherwise it defaults to $\frac{10}{n}\sqrt{n}$.

`rep_num` The amount of subsamples of the multi-omics data set used to estimate the variability of the network under the given hyperparameters setting. Defaults to 20.

`max_iter` The greatest number of times the algorithm is allowed to choose a new best λ_w. Defaults to 10.

`old_sampling` Perform the same subsampling `xstars()` would if set to TRUE. Makes a difference with bigger data sets, where computing a correlation matrix could take significantly longer. Defaults to FALSE.

`light` Do not store the "merged" matrixes recording average variability of each edge, making the algorithm more memory efficient, if set to TRUE. Defaults to TRUE.

`ebic_gamma` The γ tuning parameter for *eBIC* selection, to set between 0 and 1. When set to 0 one has the standard *BIC*. Defaults to 0.5.

`verbose` Print information regarding the network building and the network selection processes.

Details

When using `bs()`, first, `coglasso()` estimates multiple multi-omics networks with the algorithm *collaborative graphical lasso*, one for each combination of input values for the hyperparameters λ_w, λ_b and c. Then, `select_coglasso()` selects the best combination of hyperparameters given to `coglasso()` according to the selected model selection method. The three availble options that can be set for the argument `method` are "xstars", "xestars" and "ebic". For more information on these selection methods, visit the help page of `select_coglasso()`.

Value

`bs()` returns an object of `S3` class `select_coglasso` containing several elements. The most important is probably `sel_adj`, the adjacency matrix of the selected network. Some output elements depend on the chosen model selection method.
These elements are always returned, and they are the result of network estimation with `coglasso()`:

- `loglik` is a numerical vector containing the *log* likelihoods of all the estimated networks.
- `density` is a numerical vector containing a measure of the density of all the estimated networks.
- `df` is an integer vector containing the degrees of freedom of all the estimated networks.
- `convergence` is a binary vector containing whether a network was successfully estimated for the given combination of hyperparameters or not.
- `path` is a list containing the adjacency matrices of all the estimated networks.
- `icov` is a list containing the inverse covariance matrices of all the estimated networks.

- nexploded is the number of combinations of hyperparameters for which coglasso() failed to converge.
- data is the input multi-omics data set.
- hpars is the ordered table of all the combinations of hyperparameters given as input to bs(), with $\alpha(\lambda_w + \lambda_b)$ being the key to sort rows.
- lambda_w, lambda_b, and c are numerical vectors with, respectively, all the λ_w, λ_b, and c values bs() used.
- p is the vector with the number of variables for each omic layer of the data set.
- D is the number of omics layers in the data set.
- cov optional, returned when cov_output is TRUE, is a list containing the variance-covariance matrices of all the estimated networks.

These elements are returned by all selection methods available:

- sel_index_c, sel_index_lw and sel_index_lb are the indexes of the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- sel_c, sel_lambda_w and sel_lambda_b are the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- sel_adj is the adjacency matrix of the final selected network.
- sel_density is the density of the final selected network.
- sel_icov is the inverse covariance matrix of the final selected network.
- call is the matched call.
- method is the chosen model selection method.

These are the additional elements returned when choosing "xestars":

- opt_adj is a list of the adjacency matrices finally selected for each c parameter explored.
- opt_variability is a numerical vector containing the variabilities associated to the adjacency matrices in opt_adj.
- opt_index_lw and opt_index_lb are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- opt_lambda_w and opt_lambda_b are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.
- merge_lw and merge_lb are returned only if light is set to FALSE. They are lists with as many elements as the number of c parameters explored. Every element is a "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and the selected λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.

These are the additional elements returned when choosing "xstars":

- merge_lw and merge_lb are lists with as many elements as the number of c parameters explored. Every element is in turn a list of as many matrices as the number of λ_w (or λ_b) values explored. Each matrix is the "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.

- `variability_lw` and `variability_lb` are lists with as many elements as the number of c parameters explored. Every element is a numeric vector of as many items as the number of λ_w (or λ_b) values explored. Each item is the variability of the network estimated for those specific c and λ_w (or λ_b) values in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.
- `opt_adj` is a list of the adjacency matrices finally selected for each c parameter explored.
- `opt_variability` is a numerical vector containing the variabilities associated to the adjacency matrices in `opt_adj`.
- `opt_index_lw` and `opt_index_lb` are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- `opt_lambda_w` and `opt_lambda_b` are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.

These are the additional elements returned when choosing "ebic":

- `ebic_scores` is a numerical vector containing the eBIC scores for all the hyperparameter combination.

Examples

```
# Suggested usage: give the input data set, set the values for `p` and the
# number of hyperparameters to explore (to choose how extensively to explore
# the possible hyperparameters). Then, let the default behavior do the rest:

sel_mo_net <- bs(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3,
                 nlambda_b = 3, nc = 3, verbose = FALSE)
```

`coglasso` *Estimate networks from a multi-omics data set*

Description

`coglasso()` estimates multiple multi-omics networks with the algorithm *collaborative graphical lasso*, one for each combination of input values for the hyperparameters λ_w, λ_b and c.

Usage

```
coglasso(
  data,
  p = NULL,
  pX = lifecycle::deprecated(),
  lambda_w = NULL,
  lambda_b = NULL,
  c = NULL,
  nlambda_w = NULL,
  nlambda_b = NULL,
```

```
  nc = NULL,
  lambda_w_max = NULL,
  lambda_b_max = NULL,
  c_max = NULL,
  lambda_w_min_ratio = NULL,
  lambda_b_min_ratio = NULL,
  c_min_ratio = NULL,
  cov_output = FALSE,
  verbose = TRUE
)
```

Arguments

data	The input multi-omics data set. Rows should be samples, columns should be variables. Variables should be grouped by their assay (e.g. transcripts first, then metabolites). data is a required parameter.
p	A vector with with the number of variables for each omic layer of the data set (e.g. the number of transcripts, metabolites etc.), in the same order the layers have in the data set. If given a single number, coglasso() assumes that the total of data sets is two, and that the number given is the dimension of the first one.
pX	**[Deprecated]** pX is no longer supported. Please use p.
lambda_w	A vector of values for the parameter λ_w, the penalization parameter for the "within" interactions. Overrides nlambda_w.
lambda_b	A vector of values for the parameter λ_b, the penalization parameter for the "between" interactions. Overrides nlambda_b.
c	A vector of values for the parameter c, the weight given to collaboration. Overrides nc.
nlambda_w	The number of requested λ_w parameters to explore. A sequence of size nlambda_w of λ_w parameters will be generated. Defaults to 8. Ignored when lambda_w is set by the user.
nlambda_b	The number of requested λ_b parameters to explore. A sequence of size nlambda_b of λ_b parameters will be generated. Defaults to 8. Ignored when lambda_b is set by the user.
nc	The number of requested c parameters to explore. A sequence of size nc of c parameters will be generated. Defaults to 8. Ignored when c is set by the user.
lambda_w_max	The greatest generated λ_w. By default it is computed with a data-driven approach. Ignored when lambda_w is set by the user.
lambda_b_max	The greatest generated λ_b. By default it is computed with a data-driven approach. Ignored when lambda_b is set by the user.
c_max	The greatest generated c. Defaults to 10. Ignored when c is set by the user.
lambda_w_min_ratio	The ratio of the smallest generated λ_w over the greatest generated λ_w. Defaults to 0.1. Ignored when lambda_w is set by the user.
lambda_b_min_ratio	The ratio of the smallest generated λ_b over the greatest generated λ_b. Defaults to 0.1. Ignored when lambda_b is set by the user.

c_min_ratio	The ratio of the smallest generated c over the greatest generated c. Defaults to 0.1. Ignored when c is set by the user.
cov_output	Add the estimated variance-covariance matrix to the output.
verbose	Print information regarding current coglasso run on the console.

Value

coglasso() returns an object of S3 class coglasso, that has the following elements:

- loglik is a numerical vector containing the log likelihoods of all the estimated networks.
- density is a numerical vector containing a measure of the density of all the estimated networks.
- df is an integer vector containing the degrees of freedom of all the estimated networks.
- convergence is a binary vector containing whether a network was successfully estimated for the given combination of hyperparameters or not.
- path is a list containing the adjacency matrices of all the estimated networks.
- icov is a list containing the inverse covariance matrices of all the estimated networks.
- nexploded is the number of combinations of hyperparameters for which coglasso() failed to converge.
- data is the input multi-omics data set.
- hpars is the ordered table of all the combinations of hyperparameters given as input to coglasso(), with $\alpha(\lambda_w + \lambda_b)$ being the key to sort rows.
- lambda_w is a numerical vector with all the λ_w values coglasso() used.
- lambda_b is a numerical vector with all the λ_b values coglasso() used.
- c is a numerical vector with all the c values coglasso() used.
- p is the vector with the number of variables for each omic layer of the data set.
- D is the number of omics layers in the data set.
- cov optional, returned when cov_output is TRUE, is a list containing the variance-covariance matrices of all the estimated networks.
- call is the matched call.

Examples

```
# Typical usage: set the number of hyperparameters to explore
cg <- coglasso(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3,
               nlambda_b = 3, nc = 3, verbose = FALSE)

# Model selection using eXtended Efficient StARS, takes less than five seconds
sel_cg_xestars <- select_coglasso(cg, method = "xestars", verbose = FALSE)
```

get_network *Extract a* coglasso *network*

Description

get_network() extracts the selected network from a select_coglasso object, or a different specific one from either a select_coglasso or a coglasso object when specifying the optional parameters.

Usage

```
get_network(sel_cg_obj, index_c = NULL, index_lw = NULL, index_lb = NULL)
```

Arguments

sel_cg_obj	The object of S3 class select_coglasso or of S3 class coglasso.
index_c	The index of the c value different from the one selected by model selection. To set only if the desired network is not the selected one.
index_lw	The index of the λ_w value of the chosen non-optimal network. To set only if the desired network is not the selected one.
index_lb	The index of the λ_b value of the chosen non-optimal network. To set only if the desired network is not the selected one.

Details

If the input is a coglasso object, it is necessary to specify all the indexes to extract a selected network.
If the input is a select_coglasso object, it extracts by default the selected network. If the selection method was "ebic", and you want to extract a different network than the selected one, specify all indexes. Otherwise, if the objective is to extract the optimal network for a specific c value different than the selected one, set index_c to your chosen one. Also here it is possible to extract a specific non-optimal network by setting all the indexes to the chosen ones.

Value

get_network() returns the selected network, in the form of an object of class igraph.

Examples

```
sel_cg <- bs(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3, nlambda_b = 3,
                nc = 3, verbose = FALSE)
sel_net <- get_network(sel_cg)
# Could even plot the selected network with plot(sel_net), but then it would
# plot an unnotated network, better to directly plot(sel_cg).
print(sel_net)
```

`multi_omics_sd` *Multi-omics dataset of sleep deprivation in mouse*

Description

A dataset containing transcript and metabolite values analysed in Albanese et al. 2023, subset of the multi-omics data set published in Jan, M., Gobet, N., Diessler, S. et al. A multi-omics digital research object for the genetics of sleep regulation. Sci Data 6, 258 (2019).

`multi_omics_sd_small` is a smaller version, limited to the transcript Cirbp and the transcripts and metabolites belonging to its neighborhood as described in Albanese et al. 2023

`multi_omics_sd_micro` is a minimal version with Cirbp and a selection of its neighborhood.

Usage

```
multi_omics_sd

multi_omics_sd_small

multi_omics_sd_micro
```

Format

`multi_omics_sd`**:**

A data frame with 30 rows and 238 variables (162 transcripts and 76 metabolites):

Plin4 to Tfrc log2 CPM values of 162 transcripts in mouse cortex under sleep deprivation (-4.52–10.46)

Ala to SM C24:1 abundance values of 76 metabolites (0.02–1112.67)

`multi_omics_sd_small`**:**

A data frame with 30 rows and 19 variables (14 transcripts and 5 metabolites)

Cirbp to Stip1 log2 CPM values of 14 transcripts in mouse cortex under sleep deprivation (4.24–9.31)

Phe to PC ae C32:2 Abundance values of 5 metabolites (0.17–145.33)

`multi_omics_sd_micro`**:**

A data frame with 30 rows and 6 variables (4 transcripts and 2 metabolites)

Cirbp to Dnajb11 log2 CPM values of 4 transcripts in mouse cortex under sleep deprivation (4.78–9.31)

Trp to PC aa C36:3 Abundance values of 2 metabolites (58.80–145.33)

Source

Jan, M., Gobet, N., Diessler, S. et al. A multi-omics digital research object for the genetics of sleep regulation. Sci Data 6, 258 (2019) doi:10.1038/s415970190171x

Figshare folder of the original manuscript: https://figshare.com/articles/dataset/Input_data_for_systems_genetics_of_sleep_regulation/7797434

plot.select_coglasso *Plot selected* coglasso *networks*

Description

plot.select_coglasso() creates an annotated plot of a coglasso selected network from an object of S3 class select_coglasso. Variables from different data sets will have different color coding. To plot the network, it's enough to use plot() call on the select_coglasso object.

plot.coglasso() has the same functioning as select_coglasso.plot(), but from an object of S3 class coglasso. In this case, it is compulsory to specify index_c, index_lw, and index_lb.

Usage

```
## S3 method for class 'select_coglasso'
plot(
  x,
  index_c = NULL,
  index_lw = NULL,
  index_lb = NULL,
  node_labels = TRUE,
  hide_isolated = TRUE,
  ...
)

## S3 method for class 'coglasso'
plot(
  x,
  index_c,
  index_lw,
  index_lb,
  node_labels = TRUE,
  hide_isolated = TRUE,
  ...
)
```

Arguments

x	The object of S3 class select_coglasso.
index_c	The index of the c value different from the one selected by model selection. To set only if the desired network is not the selected one.
index_lw	The index of the λ_w value of the chosen non-optimal network. To set only if the desired network is not the selected one.
index_lb	The index of the λ_b value of the chosen non-optimal network. To set only if the desired network is not the selected one.
node_labels	Show node names in the network. Defaults to TRUE.

hide_isolated	Hide nodes that are not connected to any other node. Defaults to TRUE.
...	System required, not used here.

Details

If the input is a coglasso object, it is necessary to specify all the indexes to extract a selected network.
If the input is a select_coglasso object, it extracts by default the selected network. If the selection method was "ebic", and you want to extract a different network than the selected one, specify all indexes. Otherwise, if the objective is to extract the optimal network for a specific c value different than the selected one, set index_c to your chosen one. Also here it is possible to extract a specific non-optimal network by setting all the indexes to the chosen ones.

Value

Returns NULL, invisibly.

See Also

get_network() to understand what it means to select a specific network with index_c, index_lw, and index_lb.

Examples

```
sel_cg <- bs(multi_omics_sd_small, p = c(14, 5), nlambda_w = 15, nlambda_b = 15,
             nc = 3, lambda_w_min_ratio = 0.6, verbose = FALSE)
plot(sel_cg)
```

---

select_coglasso *Select the best* coglasso *network*

Description

select_coglasso() selects the best combination of hyperparameters given to coglasso() according to the selected model selection method. The three availble options that can be set for the argument method are "xstars", "xestars" and "ebic".

Usage

```
select_coglasso(
  coglasso_obj,
  method = "xestars",
  stars_thresh = 0.1,
  stars_subsample_ratio = NULL,
  rep_num = 20,
```

```
  max_iter = 10,
  old_sampling = FALSE,
  light = TRUE,
  ebic_gamma = 0.5,
  verbose = TRUE
)
```

Arguments

coglasso_obj	The object of S3 class coglasso returned by coglasso().
method	The model selection method to select the best combination of hyperparameters. The available options are "xstars", "xestars" and "eBIC". Defaults to "xestars".
stars_thresh	The threshold set for variability of the explored networks at each iteration of the algorithm. The λ_w or the λ_b associated to the most stable network before the threshold is overcome is selected.
stars_subsample_ratio	The proportion of samples in the multi-omics data set to be randomly subsampled to estimate the variability of the network under the given hyperparameters setting. Defaults to 80% when the number of samples is smaller than 144, otherwise it defaults to $\frac{10}{n}\sqrt{n}$.
rep_num	The amount of subsamples of the multi-omics data set used to estimate the variability of the network under the given hyperparameters setting. Defaults to 20.
max_iter	The greatest number of times the algorithm is allowed to choose a new best λ_w. Defaults to 10.
old_sampling	Perform the same subsampling xstars() would if set to TRUE. Makes a difference with bigger data sets, where computing a correlation matrix could take significantly longer. Defaults to FALSE.
light	Do not store the "merged" matrixes recording average variability of each edge, making the algorithm more memory efficient, if set to TRUE. Defaults to TRUE.
ebic_gamma	The γ tuning parameter for *eBIC* selection, to set between 0 and 1. When set to 0 one has the standard *BIC*. Defaults to 0.5.
verbose	Print information regarding the progress of the selection procedure on the console.

Details

select_coglasso() provides three model selection strategies:

- "xstars" uses *eXtended StARS* (*XStARS*) selecting the most stable, yet sparse network. Stability is computed upon network estimation from multiple subsamples of the multi-omics data set, allowing repetition. Subsamples are collected for a fixed amount of times (rep_num), and with a fixed proportion of the total number of samples (stars_subsample_ratio). See xstars() for more information on the methodology.
- "xestars" uses *eXtended Efficient StARS* (*XEStARS*), a significantly faster and memory-effcient version of *XStARS*. It could produce marginally different results to "xstars" due to a different sampling strategy. See xestars() for more information on the methodology.

- "ebic" uses the *extended Bayesian Information Criterion* (*eBIC*) selecting the network that minimizes it. `gamma` sets the wait given to the extended component, turning the model selection method to the standard *BIC* if set to 0.

Value

`select_coglasso()` returns an object of `S3` class `select_coglasso` containing the results of the selection procedure, built upon an object of `S3` class `coglasso`. Some output elements depend on the chosen model selection method.
These elements are returned by all methods:

- ... are the same elements returned by `coglasso()`.
- `sel_index_c`, `sel_index_lw` and `sel_index_lb` are the indexes of the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_c`, `sel_lambda_w` and `sel_lambda_b` are the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_adj` is the adjacency matrix of the final selected network.
- `sel_density` is the density of the final selected network.
- `sel_icov` is the inverse covariance matrix of the final selected network.
- `call` is the matched call.
- `method` is the chosen model selection method.

These are the additional elements returned when choosing "xestars":

- `opt_adj` is a list of the adjacency matrices finally selected for each c parameter explored.
- `opt_variability` is a numerical vector containing the variabilities associated to the adjacency matrices in `opt_adj`.
- `opt_index_lw` and `opt_index_lb` are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- `opt_lambda_w` and `opt_lambda_b` are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.
- `merge_lw` and `merge_lb` are returned only if `light` is set to FALSE. They are lists with as many elements as the number of c parameters explored. Every element is a "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and the selected λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.

These are the additional elements returned when choosing "xstars":

- `merge_lw` and `merge_lb` are lists with as many elements as the number of c parameters explored. Every element is in turn a list of as many matrices as the number of λ_w (or λ_b) values explored. Each matrix is the "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.

- variability_lw and variability_lb are lists with as many elements as the number of c parameters explored. Every element is a numeric vector of as many items as the number of λ_w (or λ_b) values explored. Each item is the variability of the network estimated for those specific c and λ_w (or λ_b) values in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.
- opt_adj is a list of the adjacency matrices finally selected for each c parameter explored.
- opt_variability is a numerical vector containing the variabilities associated to the adjacency matrices in opt_adj.
- opt_index_lw and opt_index_lb are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- opt_lambda_w and opt_lambda_b are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.

These are the additional elements returned when choosing "ebic":

- ebic_scores is a numerical vector containing the eBIC scores for all the hyperparameter combination.

Examples

```
cg <- coglasso(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3,
               nlambda_b = 3, nc = 3, verbose = FALSE)
# Using eXtended Efficient StARS, takes less than five seconds
sel_cg_xestars <- select_coglasso(cg, method = "xestars", verbose = FALSE)

# Using eXtended StARS, takes around a minute
sel_cg_xstars <- select_coglasso(cg, method = "xstars", verbose = FALSE)

# Using eBIC
sel_cg_ebic <- select_coglasso(cg, method = "ebic", verbose = FALSE)
```

---

xestars *Efficient stability selection of the best* coglasso *network*

Description

xestars() provides a more efficient and lighter implementation than xstars() to select the combination of hyperparameters given to coglasso() yielding the most stable, yet sparse network. Stability is computed upon network estimation from multiple subsamples of the multi-omics data set, allowing repetition. Subsamples are collected for a fixed amount of times (rep_num), and with a fixed proportion of the total number of samples (stars_subsample_ratio).

Usage

```
xestars(
  coglasso_obj,
  stars_thresh = 0.1,
  stars_subsample_ratio = NULL,
  rep_num = 20,
  max_iter = 10,
  old_sampling = FALSE,
  light = TRUE,
  verbose = TRUE
)
```

Arguments

coglasso_obj	The object of S3 class coglasso returned by coglasso().
stars_thresh	The threshold set for variability of the explored networks at each iteration of the algorithm. The λ_w or the λ_b associated to the most stable network before the threshold is overcome is selected.
stars_subsample_ratio	The proportion of samples in the multi-omics data set to be randomly subsampled to estimate the variability of the network under the given hyperparameters setting. Defaults to 80% when the number of samples is smaller than 144, otherwise it defaults to $\frac{10}{n}\sqrt{n}$.
rep_num	The amount of subsamples of the multi-omics data set used to estimate the variability of the network under the given hyperparameters setting. Defaults to 20.
max_iter	The greatest number of times the algorithm is allowed to choose a new best λ_w. Defaults to 10.
old_sampling	Perform the same subsampling xstars() would if set to TRUE. Makes a difference with bigger data sets, where computing a correlation matrix could take significantly longer. Defaults to FALSE.
light	Do not store the "merged" matrixes recording average variability of each edge, making the algorithm more memory efficient, if set to TRUE. Defaults to TRUE.
verbose	Print information regarding the progress of the selection procedure on the console.

Details

eXtended Efficient StARS (*XEStARS*) is a more efficient and memory-light version of *XStARS*, the adaptation for *collaborative graphical regression* of the method published by Liu, H. *et al.* (2010): Stability Approach to Regularization Selection (StARS). *StARS* was developed for network estimation regulated by a single penalty parameter, while *collaborative graphical lasso* needs to explore three different hyperparameters. In particular, two of these are penalty parameters with a direct influence on network sparsity, hence on stability. For every c parameter, xestars() explores one of the two penalty parameters (λ_w or λ_b), keeping the other one fixed at its previous best estimate, using the normal, one-dimentional *StARS* approach, until finding the best couple. What makes it more efficient than xstars() is that the stability check that in the original algorithm (even in the original *StARS*) is performed for every λ_w or λ_b value, is implemented here as a *stopping criterion*.

This reduces sensibly the number of iterations before convergence. It then selects the c parameter for which the best (λ_w, λ_b) couple yielded the most stable, yet sparse network.
The original *XStARS* computes a new subsampling for every time the algorithm switches from optimizing the twoλ_w and λ_b, and for every c. This does not allow to compare the hyperparameters on an equal ground, and can slow the selection down with bigger data set or a larger hyperparameter space. To allow a fairer (and faster) comparison among different optimizations, the `old_sampling` parameter has been implemented. If set to TRUE, the subsampling is the same one `xstars()` would perform. Otherwise the subsampling is performed at the beginning of the algorithm once and for all its iterations.
To allow `xestars()` to be more memory light, the `light` parameter has been implemented. If set to TRUE and the "merged" matrixes traditionally returned by both *StARS* and *XStARS* are not returned.

Value

`xestars()` returns an object of `S3` class `select_coglasso` containing the results of the selection procedure, built upon the object of `S3` class `coglasso` returned by `coglasso()`.

- ... are the same elements returned by `coglasso()`.
- `opt_adj` is a list of the adjacency matrices finally selected for each c parameter explored.
- `opt_variability` is a numerical vector containing the variabilities associated to the adjacency matrices in `opt_adj`.
- `opt_index_lw` and `opt_index_lb` are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- `opt_lambda_w` and `opt_lambda_b` are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.
- `sel_index_c`, `sel_index_lw` and `sel_index_lb` are the indexes of the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_c`, `sel_lambda_w` and `sel_lambda_b` are the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_adj` is the adjacency matrix of the final selected network.
- `sel_density` is the density of the final selected network.
- `sel_icov` is the inverse covariance matrix of the final selected network.
- `call` is the matched call.
- `method` is the chosen model selection method. Here, it is "xestars".
- `merge_lw` and `merge_lb` are returned only if `light` is set to FALSE. They are lists with as many elements as the number of c parameters explored. Every element is a "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and the selected λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.

Examples

```
cg <- coglasso(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3,
               nlambda_b = 3, nc = 3, verbose = FALSE)
```

```
# Takes less than five seconds
sel_cg <- xestars(cg, verbose = FALSE)
```

`xstars` *Stability selection of the best* `coglasso` *network*

Description

`xstars()` selects the combination of hyperparameters given to `coglasso()` yielding the most stable, yet sparse network. Stability is computed upon network estimation from multiple subsamples of the multi-omics data set, allowing repetition. Subsamples are collected for a fixed amount of times (`rep_num`), and with a fixed proportion of the total number of samples (`stars_subsample_ratio`).

Usage

```
xstars(
  coglasso_obj,
  stars_thresh = 0.1,
  stars_subsample_ratio = NULL,
  rep_num = 20,
  max_iter = 10,
  verbose = TRUE
)
```

Arguments

`coglasso_obj`	The object of `S3` class `coglasso` returned by `coglasso()`.
`stars_thresh`	The threshold set for variability of the explored networks at each iteration of the algorithm. The λ_w or the λ_b associated to the most stable network before the threshold is overcome is selected.
`stars_subsample_ratio`	The proportion of samples in the multi-omics data set to be randomly subsampled to estimate the variability of the network under the given hyperparameters setting. Defaults to 80% when the number of samples is smaller than 144, otherwise it defaults to $\frac{10}{n}\sqrt{n}$.
`rep_num`	The amount of subsamples of the multi-omics data set used to estimate the variability of the network under the given hyperparameters setting. Defaults to 20.
`max_iter`	The greatest number of times the algorithm is allowed to choose a new best λ_w. Defaults to 10.
`verbose`	Print information regarding the progress of the selection procedure on the console.

Details

eXtended StARS (*XStARS*) is an adaptation for *collaborative graphical regression* of the method published by Liu, H. *et al.* (2010): Stability Approach to Regularization Selection (StARS). *StARS* was developed for network estimation regulated by a single penalty parameter, while *collaborative graphical lasso* needs to explore three different hyperparameters. In particular, two of these are penalty parameters with a direct influence on network sparsity, hence on stability. For every c parameter, `xstars()` explores one of the two penalty parameters (λ_w or λ_b), keeping the other one fixed at its previous best estimate, using the normal, one-dimentional *StARS* approach, until finding the best couple. It then selects the c parameter for which the best (λ_w, λ_b) couple yielded the most stable, yet sparse network.

Value

`xstars()` returns an object of `S3` class `select_coglasso` containing the results of the selection procedure, built upon the object of `S3` class `coglasso` returned by `coglasso()`.

- ... are the same elements returned by `coglasso()`.
- `merge_lw` and `merge_lb` are lists with as many elements as the number of c parameters explored. Every element is in turn a list of as many matrices as the number of λ_w (or λ_b) values explored. Each matrix is the "merged" adjacency matrix, the average of all the adjacency matrices estimated for those specific c and λ_w (or λ_b) values across all the subsampling in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.
- `variability_lw` and `variability_lb` are lists with as many elements as the number of c parameters explored. Every element is a numeric vector of as many items as the number of λ_w (or λ_b) values explored. Each item is the variability of the network estimated for those specific c and λ_w (or λ_b) values in the last path explored before convergence, the one when the final combination of λ_w and λ_b is selected for the given c value.
- `opt_adj` is a list of the adjacency matrices finally selected for each c parameter explored.
- `opt_variability` is a numerical vector containing the variabilities associated to the adjacency matrices in `opt_adj`.
- `opt_index_lw` and `opt_index_lb` are integer vectors containing the index of the selected λ_ws (or λ_bs) for each c parameters explored.
- `opt_lambda_w` and `opt_lambda_b` are vectors containing the selected λ_ws (or λ_bs) for each c parameters explored.
- `sel_index_c`, `sel_index_lw` and `sel_index_lb` are the indexes of the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_c`, `sel_lambda_w` and `sel_lambda_b` are the final selected parameters c, λ_w and λ_b leading to the most stable sparse network.
- `sel_adj` is the adjacency matrix of the final selected network.
- `sel_density` is the density of the final selected network.
- `sel_icov` is the inverse covariance matrix of the final selected network.
- `call` is the matched call.
- `method` is the chosen model selection method. Here, it is "xstars".

Examples

```
cg <- coglasso(multi_omics_sd_micro, p = c(4, 2), nlambda_w = 3,
               nlambda_b = 3, nc = 3, verbose = FALSE)

# Takes around one minute
sel_cg <- xstars(cg, verbose = FALSE)
```

Index

∗ **datasets**
multi_omics_sd, 10

bs, 2

coglasso, 6
coglasso(), *2*, *4*, *14*, *17*, *19*

get_network, 9
get_network(), *12*

multi_omics_sd, 10
multi_omics_sd_micro *(*multi_omics_sd*)*, 10
multi_omics_sd_small *(*multi_omics_sd*)*, 10

plot.coglasso *(*plot.select_coglasso*)*, 11
plot.select_coglasso, 11

select_coglasso, 12
select_coglasso(), *2*, *4*

xestars, 15
xestars(), *13*
xstars, 18
xstars(), *13*