

Gleesso

December 7, 2017

batch_converter	<i>batch community converter</i>
-----------------	----------------------------------

Description

convert community (give the same indice) of a batch of graphs to the community of a reference graph

Usage

batch_converter(graph_batch, graph_ref)

Arguments

graph_batch : a list of graph to convert
graph_ref : the reference used for conversion Putting graph reference at the list of graph

Details

#####

community_converter	<i>##### Community converter function take to two nodes table as argument and give a translation of each community to the other graph based on jacquart distance If I may, it is automated community translation</i>
---------------------	--

Description

Community converter function take to two nodes table as argument and give a translation of each community to the other graph based on jacquart distance If I may, it is automated community translation

Usage

```
community_converter(nodes_graph1, nodes_graph2, join_type = "inner")
```

Arguments

nodes_graph1 is the table of nodes with the walktrap column properly filled
 nodes_graph2 is the table of nodes of the second graph

Compute_community_abondance

compute community abundances

Description

Compute the sum of abundance of species in each community given the abundance matrix by species and the graph community. Also compute the p-value of difference of abundance given a contrast vector with 2 classes (2 classes)

Usage

```
Compute_community_abondance(Nodes, abundance, species_taxo, contrast = NULL,  
  community_kind = "walktrap_community")
```

Arguments

Nodes: The network node table with the a community attribution column
 abundance: The Metagenomic species abundance table
 species_taxo: The Metagenomic species taxonomy table
 contrast: a boolean vector to form two group of samples. for each community the rank test difference of abundance p-value is calculated between the two groups.
 community kind: the algorithm of used to compute the community : "spinglass_community", "walktrap_community"

Value

a table of community abundance and composition

Compute_graph

Compute Glasso

Description

Compute the Glasso model from the MGS abundances on individuals with a true value in the contrast vector.

Usage

```
Compute_graph(MGS_abundance, contrast_vector, fout,
  abundance_treshold = 10^-7, occurrence_treshold = 0.05, nlambda = 20,
  lambda.min.ratio = 0.1, lambda = NULL, rep.num = 20)
```

Arguments

contrast_vector

: a boolean vector to select a subset of the cohort (the model will be inferred on samples with a TRUE value)

MGS_abundance: The Metagenomic species abundance table

fout: where to save the model object

community: Should the community structure be calculated?

nlambda: Number of regularisation parameter that will be tested (see huge::huge() documentation)

lambda.min.ratio:

the smallest value of lambda as a fraction of its maximum (see huge::huge() documentation)

occurrence_treshold:

minimum fraction of samples where a species must be present to be taken into account in the analysis

abundance_threshold:

minimum mean abundances for a species to be included in the analysis

rep.num:

Number of subsampling to compute the edge stability with "StarS" (see huge::huge.select documentation)

lambda:

A sequence of regularisation parameter. If not null, it will override the automatic computation of the lambda sequence (with nlambda and lambda.min.ratio)

concordance_table	<i>concordance_table</i>
-------------------	--------------------------

Description

Enable one to assess if community found in diverse cohort are the same Used to generate the alluvial plot

Usage

```
concordance_table(nlist, Graph_tags, join_type = "outer")
```

Arguments

nlist	: list of graphs nodes table with the walktrap_community information Available
Graph_tags	: list of graph labels
join_type	: how to join graph row (outer joins or inner join). Outer join means that all species present in at least one graph will be taken into account. Inner join means that only species present in all graphs will treated.

create_graph	<i>Create a gephi format graph from the graphical Lasso model</i>
--------------	---

Description

The Function also compute the community structure of the graph with various algorithms (betweenness community, walktrap community...) community specified by the user

Usage

```
create_graph(file_input, file_output, MGS_by_taxo_species, species_taxo,
  nspins = 20, mod_rep = 10, community = FALSE, additional_info = NULL,
  spinglass_opt = FALSE, variability_treshold = NULL)
```

Arguments

spinglass_opt	: Should the number of spin of the spinglass community be optimized?
file_input:	emplacement of the GLASSO model object
file_output:	where to save the network representation file
MGS_by_taxo_species:	The Metagenomic species abundance table
species_taxo:	The Metagenomic species taxonomy table
nspins:	number of spin for the spinglass community detection algorithm

community: Should the community structure be calculated?

additional_info: a vector or data.frame containing information to add to the nodes table of the network

variability_treshold: The maximum mean variability for graph edge presence. If null, the optimal covariance matrix will correspond to a variability of 0.05

generate_graph_from_tables

generate_graph_from_tables create a gephi graph from a Nodes table and Glasso model object

Description

generate_graph_from_tables create a gephi graph from a Nodes table and Glasso model object

Usage

```
generate_graph_from_tables(fout, nodes_viz_att, fgraph_model,
  variability_treshold = NULL)
```

Gleesso_bootstrap *bootstrap Gleesso_pipeline*

Description

Apply the bootstrap pipeline to a fraction of the cohort. A factor vector can be supplied to stratify the different samples

Usage

```
Gleesso_bootstrap(N_bootstrap, fraction, model_tag, variability_treshold,
  community_table_folder, model_folder, graph_folder, MGS_file, taxo_file,
  stratifying_vector = NULL, ...)
```

Arguments

N_bootstrap : number of different bootstrap samples that should be drawn

fraction : fraction of the initial dataset that should be drawn to form each bootstrap samples

model_tag :

stratifying_vector : a factor vector that represent a class that should be evenly scattered between bootstrap samples

... : parameters to pass to the
 Graphs_folder : folder to output all graphs and all bootstrap samples (to keep track of which individual was used in each iteration)

Gleesso_pipeline *Pipeline launcher*

Description

Launching the complete glasso analysis (data prep, graph inference, community detection, community abundances computation) Warning : folder shouldn't be indicated with an / at the end

Usage

```
Gleesso_pipeline(data_folder, MGS_file, taxo_file, model_folder, graph_folder,
  contrast_vector, tag_model, tag_graph, community = TRUE, nlambda = 20,
  lambda.min.ratio = 0.1, occurence_treshold = 0.05,
  abundance_treshold = 10^-7, variability_treshold = NULL,
  analysis_step = NULL, species_mode = TRUE)
```

Arguments

data_folder : where the community abundance table will be written
 contrast_vector : a boolean vector to select a subset of the cohort (the model will be inferred on samples with a TRUE value)
 tag_model : a tag that will be inserted in output file to recognize the model parameter
 tag_graph : a tag that will be inserted in output file to recognize the graph
 species_mode : should the graph inference be done on MGS (FALSE) or with MGS of the same species merged together (TRUE)
 MGS_file: The Metagenomic species abundance file in the RDS format
 model_folder: where to save the graphical Lasso model object (output of the spiec.easi function)
 graph_folder: where the graphical representation of the model will be saved (in the gephi format)
 community: Should the community structure be calculated?
 nlambda: Number of regularisation parameter that will be tested (see huge::huge() documentation)
 lambda.min.ratio: the smallest value of lambda as a fraction of its maximum (see huge::huge() documentation)
 occurence_treshold: minimum fraction of samples where a species must be present to be taken into account in the analysis

abundance_threshold:
 minimum mean abundances for a species to be included in the analysis

variability_treshold:
 The maximum mean variability for graph edge presence. If null, the optimal covariance matrix will correspond to a variability of 0.05

analysis_step: At which step the analysis should be started (0: from scratch, 1: model inferences, 2: save gephi network, 3: Community detection). If NULL (default), the step will be inferred from the files present in the output folders. Use analysis_step=0 to force computation from scratch.

opt_spinglass_com	<i>Computing spinglass communities and their modularity for a range of number of spin We then retrieve the optimal number of spin according to modularity</i>
-------------------	---

Description

Computing spinglass communities and their modularity for a range of number of spin We then retrieve the optimal number of spin according to modularity

Usage

```
opt_spinglass_com(con.grph, mod_rep, nspins)
```

parallel_coord_community	<i>Draw a parrallel coord graph of community belonging for different graph object. Enable one to compare and understand the stability or discrepancy between graph community</i>
--------------------------	--

Description

Draw a parrallel coord graph of community belonging for different graph object. Enable one to compare and understand the stability or discrepancy between graph community

Usage

```
parallel_coord_community(graph_node_list, Graph_tags, measure = "sum_ab",
    color_graph = 1, join_type = "inner")
```

Arguments

color_graph : index of the graph used to color the parallel coordiante plot
graph_node_list: a sequence of nodes tables with community annotated
Graph_tags: a sequence of str which are the name of nodes tables of graph_node_list
measure: the weight attributed to each CAG either "sum" of abundance or "count" of objects

+ *overload '+' operator to allow character strings concatenation*

Description

overload '+' operator to allow character strings concatenation

Usage

"+"(e1, e2)

Robust_table_community
Community attribution stability table

Description

Compute the stability of species community attribution from bootstraped graphs and community table abundance for robust attribution.

Usage

```
Robust_table_community(graphs_folder, alluvial_diagnostic_file, taxo,  
  N_alluvial = 10, join_type = "outer", stability_treshold = 0.6,  
  silhouette_treshold = 0.1)
```

Arguments

graphs_folder : folder where all graphs are placed (with the / at the end please ^^)
N_alluvial : number of graph to represent on a graph
alluvial_diagnostic : file name for the alluvial graph

Value

a list object containing the following elements: staby : table of species stability with community assignation for all graphs stab_n_taxo : table of species stability and taxonomy Robust_community_stability[...] : abundance of communities for species with a stability above the specified treshold Robust_community_stability[...]_silhou : idem but species also have a silhouette above a specified treshold

```
Silhouette_to_community
      Silhouette_to_community
```

Description

Compute the silhouette cluster metric for all species

Usage

```
Silhouette_to_community(my_dist, Nodes_with_com)
```

Arguments

`my_dist` : distance matrix of species to all species

`Nodes_with_com` : Nodes table with a walktrap_community attribution `cl_community = Nodes_with_com$walktrap_community`
`converter = as.data.frame(t(as.data.frame(strsplit(unique(paste(Nodes_with_com$walktrap_community, cl_community, sep="_")), "_")), stringAsFactors = FALSE) row.names(converter) = converter$V2 S_distdt[,1] = converter[SDist[,1], 1] S_distdt[,2] = converter[SDist[,2], 1])`

```
stability_index_converter
      #####
      stability_index function look at the walktrap community of each
      species in a list of graph then compute the number of graph where the
      species as been attributed to the same community
```

Description

```
#####
stability_index function look at the walktrap community of each species in a list of graph then compute the number of graph where the species as been attributed to the same community
```

Usage

```
stability_index_converter(graph_list, join_type = "outer")
```

Arguments

`graph_batch` : a list of graph with walktrap community converted to a ref

walktrap_distance	<i>walktrap_distance</i>
-------------------	--------------------------

Description

reproduce the distance used in the walktrap community detection algorithm

Usage

walktrap_distance(pos.graph, n_steps)

Arguments

- pos.graph: an igraph object that contain only positive edges
- n_steps: number of steps of the random walk on the graph #####

Index

+, [8](#)

batch_converter, [1](#)

community_converter, [1](#)

Compute_community_abondance, [2](#)

Compute_graph, [3](#)

concordance_table, [4](#)

create_graph, [4](#)

generate_graph_from_tables, [5](#)

Gleesso_bootstrap, [5](#)

Gleesso_pipeline, [6](#)

opt_spinglass_com, [7](#)

parrallel_coord_community, [7](#)

Robust_table_community, [8](#)

Silhouette_to_community, [9](#)

stability_index_converter, [9](#)

walktrap_distance, [10](#)