Package 'SEMgraph'

July 8, 2021

Title Network Analysis and Causal Inference Through Structural Equation Modeling

Version 1.0.3

Date 2021-07-07

Description Estimate networks and causal relationships in complex systems through Structural Equation Modeling. This package also includes functions to import, weight, manipulate, and fit biological network models within the Structural Equation Modeling framework;
Palluzzi and Grassi (2021) <arXiv:2103.08332>.

URL https://github.com/fernandoPalluzzi/SEMgraph

Depends igraph (>= 1.2.1), lavaan (>= 0.5-23), R (>= 4.0)

Imports boot (>= 1.3-25), cate (>= 1.0.4), corpcor (>= 1.6.9), dagitty (>= 0.3-0), diffusr (>= 0.1.4), flip (>= 2.5.0), gdata (>= 2.18.0), ggm (>= 2.3), GGMncv (>= 2.0.0), glmnet (>= 2.0-18), graph (>= 1.56.0), Matrix (>= 1.3-0), pbapply (>= 1.4-3), protoclust (>= 1.6.3), RcppEigen (>= 0.3.3.4.0), Rgraphviz (>= 2.22.0)

Suggests huge, pcalg, graphite, org.Hs.eg.db

License GPL-3

LazyData true

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.1.1

Maintainer Fernando Palluzzi <fernando.palluzzi@gmail.com>

Author Mario Grassi [aut],
Fernando Palluzzi [aut, cre],
Daniele Pepe [ctb]

Repository CRAN

Date/Publication 2021-07-08 06:30:02 UTC

R topics documented:

activeModule . . . 2
alsData . . . 5
ancestry . . . 6
clusterGraph . . . 7
clusterScore . . . 8
colorGraph . . . 10
corr2graph . . . 12
cplot . . . 13
extendGraph . . . 14
extractClusters . . . 16
gplot . . . 18
graph2dag . . . 19
graph2dagitty . . . 20
graph2lavaan . . . 21
kegg . . . 22
lavaan2graph . . . 23
mergeNodes . . . 24
modelSearch . . . 25
orientEdges . . . 28
pairwiseMatrix . . . 29
pathFinder . . . 30
properties . . . 32
sachs . . . 33
SEMace . . . 34
SEMbap . . . 35
SEMdag . . . 37
SEMgsa . . . 39
SEMpath . . . 41
SEMrun . . . 42
Shipley.test . . . 46
summary.GGM . . . 48
summary.RICF . . . 48
weightGraph . . . 49

Index **52**

`activeModule` *Active module identification*

Description

Uses different information flow and tree-based strategies for identifying active modules (e.g., disease modules), showing a perturbed subset of nodes and edges. Function scalability enables graph reduction at both pathway and entire interactome scales.

Usage

```
activeModule(
  graph,
  type,
  seed,
  eweight = "none",
  alpha = 0.05,
  q = 0.5,
  limit = 30000,
  ...
)
```

Arguments

graph	An igraph object.
type	Module identification method. If type = "kou", the Steiner tree algorithm will be applied. If type = "usp", the resulting graph will be the union of all significant shortest paths. If type = "rwr", the random walk with restart algorithm will be enabled. Finally, if type = "hdi", the heat diffusion algorithm is used.
seed	Either a user-defined vector containing seed node names or one among: "pvlm", "proto", or "qi", corresponding to the seed name attribute yielded by weightGraph.
eweight	Edge weight type derived from weightGraph or from user-defined distances. This option determines the weight-to-distance transform. If set to "none" (default), edge weights will be set to 1. If eweight = "kegg", repressing interactions (-1) will be set to 1 (maximum distance), neutral interactions (0) will be set to 0.5, and activating interactions (+1) will be set to 0 (minimum distance). If eweight = "zsign", all significant interactions will be set to 0 (minimum distance), while non-significant ones will be set to 1. If eweight = "pvalue", weights (p-values) will be transformed to the inverse of negative base-10 logarithm. If eweight = "custom", the algorithm will use the distance measure specified by the user as "weight" edge attribute.
alpha	Significance level to assess shortest paths significance, when type is "usp". By default, alpha = 0.05.
q	Inclusion quantile for the "rwr" and "hdi" algorithms. The higher the q, the closer the nodes to the input seeds, the smaller the output graph induced by the q-top ranking nodes. By default, q = 0.5 (i.e., the top 50% of nodes are selected).
limit	An integer value corresponding to the number of graph edges. If type = "usp", beyond this limit, multicore computation is enabled to reduce the computational burden. By default, limit = 30000.
...	Currently ignored.

Details

Graph filtering algorithms include:

1. "kou", the Steiner tree connecting a set of seed nodes, using the algorithm from Kou et al. (1981);
2. "usp", generates a subnetwork as the union of the significant (P-value < alpha) shortest paths between the seeds set;
3. "rwr", Random Walk with Restart; wrapper for random.walk of the R package diffusr;
4. "hdi", Heat Diffusion algorithm; wrapper for heat.diffusion of the R package diffusr.

Value

An active module of class igraph.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

Kou L, Markowsky G, Berman L (1981). A fast algorithm for Steiner trees. Acta Informatica, 15(2): 141-145. <https://doi.org/10.1007/BF00288961>

Simon Dirmeier (2018). diffusr: Network Diffusion Algorithms. R package version 0.1.4. <https://CRAN.R-project.org/package=diffusr/>

Examples

```
# Graph weighting
G <- weightGraph(graph = sachs$graph, data = sachs$pkc, group = sachs$group,
                 method = "r2z",
                 seed = c(0.05, 0.5, 0.5))

# RWR algorithm, seeds and edge P-values as weights
R1 <- activeModule(graph = G, type = "kou", seed = "pvlm", eweight = "pvalue")
R2 <- activeModule(graph = G, type = "kou", seed = "proto", eweight = "pvalue")
R3 <- activeModule(graph = G, type = "kou", seed = "qi", eweight = "pvalue")

# Graphs
old.par <- par(no.readonly = TRUE)
par(mfrow=c(2,2), mar=rep(2, 4))
plot(G, layout = layout.circle, main = "input graph")
box(col = "gray")
plot(R1, layout = layout.circle, main = "lm P-value (alpha = 0.05)")
box(col = "gray")
plot(R2, layout = layout.circle, main = "prototype (h = 0.5)")
box(col = "gray")
plot(R3, layout = layout.circle, main = "closeness (q = 0.5)")
box(col = "gray")
par(old.par)
```

`alsData` *Amyotrophic Lateral Sclerosis (ALS) dataset*

Description

Expression profiling through high-throughput sequencing (RNA-seq) of 139 ALS patients and 21 healthy controls (HCs), from Tam et al. (2019).

Usage

```
alsData
```

Format

alsData is a list of 4 objects:

1. "graph", ALS graph as the largest connected component of the "Amyotrophic lateral sclerosis (ALS)" pathway from KEGG database;
2. "exprs", a matrix of 160 rows (subjects) and 303 columns (genes) extracted from the original 17695. This subset includes genes from KEGG ALS signaling pathway, MAPK signaling pathway, and Steroid biosynthesis pathway, needed to run SEMgraph examples. Raw data from the GEO dataset GSE124439 (Tam et al., 2019) were pre-processed applying batch effect correction, using the sva R package (Leek et al., 2012), to remove data production center and brain area biases. Using multidimensional scaling-based clustering, ALS-specific and an HC-specific clusters were generated. Misclassified samples were blacklisted and removed from the current dataset;
3. "group", a binary group vector of 139 ALS subjects (1) and 21 healthy controls (0);
4. "details", a data.frame reporting information about included and blacklisted samples.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE124439/

References

Tam OH, Rozhkov NV, Shaw R, Kim D et al. (2019). Postmortem Cortex Samples Identify Distinct Molecular Subtypes of ALS: Retrotransposon Activation, Oxidative Stress, and Activated Glia. Cell Repprts, 29(5):1164-1177.e5. <https://doi.org/10.1016/j.celrep.2019.09.066>

Jeffrey T. Leek, W. Evan Johnson, Hilary S. Parker, Andrew E. Jaffe, and John D. Storey (2012). The sva package for removing batch effects and other unwanted variation in high-throughput experiments. Bioinformatics. Mar 15; 28(6): 882-883. <https://doi.org/10.1093/bioinformatics/bts034>

Examples

```
alsData$graph
dim(alsData$exprs)
table(alsData$group)
```

ancestry *Node ancestry utilities*

Description

Get ancestry for a collection of nodes in a graph. These functions are wrappers for the original SEMID R package.

Usage

```
ancestors(g, nodes)

descendants(g, nodes)

parents(g, nodes)

siblings(g, nodes)
```

Arguments

g	An igraph object.
nodes	the nodes in the graph of which to get the ancestry.

Value

a sorted vector of nodes.

References

Rina Foygel Barber, Mathias Drton and Luca Weihs (2019). SEMID: Identifiability of Linear Structural Equation Models. R package version 0.3.2. <https://CRAN.R-project.org/package=SEMID/>

Examples

```
# Get all ancestors
an <- V(sachs$graph)[ancestors(sachs$graph, "Erk")]; an

# Get parents
pa <- V(sachs$graph)[parents(sachs$graph, "PKC")]; pa

# Get descendants
de <- V(sachs$graph)[descendants(sachs$graph, "PKA")]; de

# Get siblings
sib <- V(sachs$graph)[siblings(sachs$graph, "PIP3")]; sib
```

clusterGraph *Topological graph clustering*

Description

Topological graph clustering methods.

Usage

```
clusterGraph(graph, type = "wtc", HM = "none", size = 5, verbose = FALSE, ...)
```

Arguments

graph	An igraph object.
type	Topological clustering methods. If type = "tahc", network modules are generated using the tree agglomerative hierarchical clustering method (Yu et al., 2015). Other non-tree clustering methods from igraph package include: "wtc" (default value; walktrap community structure with short random walks), "ebc" (edge betweeness clustering), "fgc" (fast greedy method), "lbc" (label propagation method), "lec" (leading eigenvector method), "loc" (multi-level optimization), "opc" (optimal communiy structure), "sgc" (spinglass statistical mechanics).
HM	Hidden model type. Enables the visualization of the hidden model. If set to "none" (default), no HM is visualized. For each defined hidden module: (i) if HM = "LV", a latent variable (LV) will be defined as common unknown cause acting on cluster nodes; (ii) if HM = "CV", cluster nodes will be considered as regressors of a latent composite variable (CV); (iii) if HM = "UV", an unmeasured variable (UV) is defined, where source nodes of the module (i.e., in-degree = 0) act as common regressors influencing the other nodes via an unmeasured variable (see also clusterScore).
size	Minimum number of nodes per module. By default, a minimum number of 5 nodes is required.
verbose	A logical value. If FALSE (default), the processed graphs will not be plotted to screen, saving execution time (they will be returned in output anyway).
...	Currently ignored.

Value

If HM is not "none" a list of 3 objects is returned:

1. "gHM", subgraph containing hidden modules as an igraph object;
2. "membership", cluster membership vector for each node;
3. "gHC", the list of modules as igraph objects.

If HM is "none", only the cluster membership vector is returned.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Fortunato S, Hric D. Community detection in networks: A user guide (2016). Phys Rep; 659: 1-44. <https://dx.doi.org/10.1016/j.physrep.2016.09.002>

Yu M, Hillebrand A, Tewarie P, Meier J, van Dijk B, Van Mieghem P, Stam CJ (2015). Hierarchical clustering in minimum spanning trees. Chaos 25(2): 023107. <https://doi.org/10.1063/1.4908014>

See Also

clusterScore, cplot

Examples

```
# Clustering ALS graph with WTC method and LV model
G <- properties(alsData$graph)[[1]]
membership <- clusterGraph(graph = G, type = "wtc", HM = "LV", verbose = TRUE)
```

---

clusterScore *Module scoring*

Description

Generate factor scores, principal component scores, or projection scores of latent, composite, and unmeasured variable modules, respectively, and fit them in a SEM with exogenous group effect.

Usage

```
clusterScore(
  graph,
  data,
  group,
  HM = "LV",
  type = "wtc",
  size = 5,
  verbose = FALSE,
  ...
)
```

Arguments

graph	An igraph object.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes.
group	A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects.
HM	Hidden model type. For each defined hidden module: (i) if HM = "LV", a latent variable (LV) will be defined as common unknown cause acting on cluster nodes; (ii) if HM = "CV", cluster nodes will be considered as regressors of a latent composite variable (CV); (iii) if HM = "UV", an unmeasured variable (UV) model will be generated for each module, where source nodes (i.e., in-degree = 0) act as common regressors influencing the other nodes via an unmeasured variable.
type	Graph clustering method. If type = "tahc", network modules are generated using the tree agglomerative hierarchical clustering method (Yu et al., 2015). Other non-tree clustering methods from igraph package include: "wtc" (default value; walktrap community structure with short random walks), "ebc" (edge betweeness clustering), "fgc" (fast greedy method), "lbc" (label propagation method), "lec" (leading eigenvector method), "loc" (multi-level optimization), "opc" (optimal communiy structure), "sgc" (spinglass statistical mechanics). By default, the "wtc" method is used.
size	Minimum number of nodes per hidden module. By default, a minimum number of 5 nodes is required. By default, HM is set to "LV" (i.e., the latent variable model).
verbose	A logical value. If TRUE, intermediate graphs will be displayed during the execution. In addition, a condensed graph with clusters as nodes will be fitted and showed to screen (see also mergeNodes). By default, verbode = FALSE.
...	Currently ignored.

Value

A list of 3 objects:

1. "fit", hidden module fitting as a lavaan object;
2. "membership", hidden module nodes membership; clusterGraph function;
3. "dataHM", hidden module data matrix with cluster scores.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

See Also

See clusterGraph and cplot for graph clustering, and factor.analysis for factor analysis.

Examples

```
library(huge)
als.npn <- huge.npn(alsData$exprs)

C <- clusterScore(graph = alsData$graph, data = als.npn,
                  group = alsData$group,
                  HM = "LV",
                  type = "wtc",
                  verbose = TRUE)
summary(C$fit)
head(C$dataHM)
table(C$membership)
```

---

colorGraph *Vertex and edge graph coloring on the base of fitting*

Description

Add vertex and edge color atrributes to an igraph object, based on a fitting results data.frame generated by SEMrun.

Usage

```
colorGraph(
  est,
  graph,
  group,
  method = "none",
  alpha = 0.05,
  vcolor = c("lightblue", "white", "pink"),
  ecolor = c("royalblue3", "gray50", "red2"),
  ewidth = c(1, 2),
  ...
)
```

Arguments

est	A data.frame of estimated parameters and p-values, derived from the fit object returned by SEMrun. As an alternative, the user may provide a "gest" or "dest" data.frame generated by SEMrun.
graph	An igraph object.

group	group A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects.
method	Multiple testing correction method. One of the values available in p.adjust. By default, method is set to "none" (i.e., no multiple test correction).
alpha	Significance level for node and edge coloring (by default, alpha = 0.05).
vcolor	A vector of three color names. The first color is given to nodes with P-value < alpha and beta < 0, the third color is given to nodes with P-value < alpha and beta > 0, and the second is given to nodes with P-value > alpha. By default, vcolor = c("lightblue","white","pink").
ecolor	A vector of three color names. The first color is given to edges with P-value < alpha and regression coefficient < 0, the third color is given to edges with P-value < alpha and regression coefficient > 0, and the second is given to edges with P-value > alpha. By default, vcolor = c("blue","gray50","red2").
ewidth	A vector of two values. The first value refers to the basic edge width (i.e., edges with P-value > alpha), while the second is given to edges with P-value < alpha. By default ewidth = c(1, 2).
...	Currently ignored.

Value

An igraph object with vertex and edge color and width attributes.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Model fitting: node perturbation
sem1 <- SEMrun(graph = alsData$graph, data = alsData$exprs,
               group = alsData$group,
               fit = 1)
est1 <- parameterEstimates(sem1$fit)

# Model fitting: edge perturbation
sem2 <- SEMrun(graph = alsData$graph, data = alsData$exprs,
               group = alsData$group,
               fit = 2)
est20 <- subset(parameterEstimates(sem2$fit), group = 1)[, -c(4, 5)]
est21 <- subset(parameterEstimates(sem2$fit), group = 2)[, -c(4, 5)]

# Graphs
g <- alsData$graph
x <- alsData$group

old.par <- par(no.readonly = TRUE)
```

```
par(mfrow=c(2,2), mar=rep(1,4))
gplot(colorGraph(est = est1, g, group = x, method = "BH"),
      main = "vertex differences")
gplot(colorGraph(est = sem2$dest, g, group = NULL),
      main = "edge differences")
gplot(colorGraph(est = est20, g, group = NULL),
      main = "edges for group = 0")
gplot(colorGraph(est = est21, g, group = NULL),
      main = "edges for group = 1")
par(old.par)
```

`corr2graph` *Correlation matrix to graph*

Description

Convert a correlation matrix to an igraph object.

Usage

```
corr2graph(R, n, type = "marg", method = "none", alpha = 0.05, ...)
```

Arguments

`R`	Correlation matrix.
`n`	Sample size (i.e., the number of subjects).
`type`	Graph building method. If `type` is either `"marg"` or `"cond"`, marginal or conditional correlation tests will be used, respectively. If `type = "mst"`, input correlations are converted to distances and a minimum spanning tree is generated from the distance matrix, using Prim's algorithm (Prim, 1957). If `type = "tmfg"`, a triangulate maximally graph is generated from the given correlation matrix (Massara et al., 2016).
`method`	Multiple testing correction method. One of the values available in `p.adjust`. By default, `method = "none"` (i.e., no multiple test correction). See `p.adjust` for other correction methods.
`alpha`	Significance level used to compute the correlation threshold. By default, `alpha = 0.05`.
`...`	Currently ignored.

Value

An igraph object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

Massara GP, Di Matteo T and Aste T (2009). Network Filtering for Big Data: Triangulated Maximally Filtered Graph. Journal of complex Networks, 5(2): 161–178. <https://doi.org/10.1093/comnet/cnw015>

Prim RC (1957). Shortest connection networks and some generalizations. Bell System Technical Journal, 36(6):1389–1401. <https://doi.org/10.1002/j.1538-7305.1957.tb01515.x>

Examples

```
# Graphs creation
C1 <- corr2graph(R = cor(log(sachs$pkc)), n = nrow(sachs$pkc),
                 type = "marg",
                 method = "BH")
C2 <- corr2graph(R = cor(log(sachs$pkc)), n = nrow(sachs$pkc),
                 type = "cond",
                 method = "BH")
C3 <- corr2graph(R = cor(log(sachs$pkc)), n = nrow(sachs$pkc),
                 type = "mst",
                 method = "BH")
C4 <- corr2graph(R = cor(log(sachs$pkc)), n = nrow(sachs$pkc),
                 type = "tmfg",
                 method = "BH")

# Graphs plots
old.par <- par(no.readonly = TRUE)
par(mfrow=c(2,2), mar= rep(2, 4))
plot(C1, layout=layout.circle, main= "marg"); box(col="gray")
plot(C2, layout=layout.circle, main= "cond"); box(col="gray")
plot(C3, layout=layout.circle, main= "mst"); box(col="gray")
plot(C4, layout=layout.circle, main= "tmfg"); box(col="gray")
par(old.par)
```

---

cplot *Subgraph mapping*

Description

Map groups of nodes onto an input graph, based on a membership vector.

Usage

```
cplot(graph, membership, l = layout.auto, map = FALSE, verbose = FALSE, ...)
```

Arguments

graph	An igraph object.
membership	Cluster membership vector for each node.
l	graph layout. One of the igraph layouts. If this argument is ignored, an automatic layout will be applied.
map	A logical value. Visualize cluster mapping over the input graph. If FALSE (default), visualization will be disabled. For large graphs, visualization may take long.
verbose	A logical value. If FALSE (default), the processed graphs will not be plotted to screen, saving execution time (they will be returned in output anyway).
...	Currently ignored.

Value

The list of clusters and cluster mapping as igraph objects.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

See Also

clusterGraph, clusterScore

Examples

```
# Clustering ALS graph with WTC method
G <- alsData$graph
membership <- clusterGraph(graph = G, type = "wtc")
cplot(G, membership, map = TRUE, verbose = FALSE)
cplot(G, membership, map = FALSE, verbose = TRUE)
```

---

extendGraph *Interactome-assisted graph extension*

Description

Extend an input directed graph, importing new interactions from a second graph. Added interactions will be chosen among those available in a given reference interactome.

Usage

```
extendGraph(g = list(), data, gnet, verbose = FALSE, ...)
```

Arguments

g	A list of two graphs as igraph objects.
data	A matrix with rows corresponding to subjects, and columns to graph nodes.
gnet	External interaction network as an igraph object. Interaction data from this network will be used to integrate additional interaction information inside the graph.
verbose	A logical value. If FALSE (default), the processed graphs will not be plotted to screen, saving execution time (they will be returned anyway).
...	Currently ignored.

Details

This function takes two input graphs: the first is the input causal model (i.e., a directed graph), and the second can be either a directed or undirected graph, providing a set of connections to be checked against the reference network and imported to the first graph. Typically, the second graph is the output of either SEMdag or SEMbap. In the former we use the new inferred causal structure stored in the dag.red object. In the latter, we use the new inferred covariance structure stored in the guu object. In both cases, new hidden directed paths and new nodes (i.e., new mediators) can be revealed.

Value

A list of 2 objects:

1. "Ug", the extended graph (union of the input graph and guv);
2. "guv", the directed subgraph added to the input graph.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

Examples

```
# Extract from graphite the "Steroid biosynthesis" pathway:

library(graphite)
humanKegg <- pathways("hsapiens", "kegg")
```

```
p <- humanKegg[["Steroid biosynthesis"]]
g <- pathwayGraph(p)
graph::nodes(g) <- gsub("ENTREZID:","",graph::nodes(g))
G <- properties(graph_from_graphnel(g))[[1]]

# Extend a graph using new inferred DAG edges:

library(huge)
als.npn <- huge.npn(alsData$exprs)

dag <- SEMdag(graph = G, data = als.npn, beta = 0.1)
ext <- extendGraph(list(dag$dag, dag$dag.red), data = als.npn, gnet = kegg)
gplot(ext$Ug)

# Extend a graph using the inferred bow-free path diagram:

bap <- SEMbap(graph = G, data = als.npn, gnet = kegg, d = 1, alpha = 0.05)
ext <- extendGraph(list(bap$bap, bap$guu), data = als.npn, gnet = kegg)
gplot(ext$Ug)

# Create a graph from correlation matrix, using KEGG as reference:

v <- which(colnames(als.npn) %in% V(G)$name)
selectedData <- als.npn[, v]
G0 <- make_empty_graph(n = ncol(selectedData))
V(G0)$name <- colnames(selectedData)

G1 <- corr2graph(R = cor(selectedData), n = nrow(selectedData),
                 type = "tmfg")
ext <- extendGraph(list(G0, G1), data = selectedData, gnet = kegg)

#Graphs
old.par <- par(no.readonly = TRUE)
par(mfrow=c(1,2), mar=rep(1,4))
plot(G1, layout = layout.circle)
plot(ext$Ug, layout = layout.circle)
par(old.par)
```

`extractClusters` *Cluster extraction utility*

Description

Extract and fit clusters from an input graph.

Usage

```
extractClusters(
  graph,
  data,
  group = NULL,
  membership = NULL,
  map = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

graph	Input network as an igraph object.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
group	A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. Group specification enables node perturbation testing. By default, group = NULL.
membership	A vector of cluster membership IDs. If NULL, clusters will be automatically generated with clusterGraph using the edge betweenness clustering ("ebc") algorithm.
map	Logical value. If TRUE, the plot of the input graph (coloured by cluster membership) will be generated along with independent module plots. If the input graph is very large, plotting could be computationally intensive (by default, map = FALSE).
verbose	Logical value. If TRUE, a plot will be showed for each cluster.
...	Currently ignored.

Value

List of clusters as igraph objects and fitting results for each cluster as a lavaan object.

Author(s)

Fernando Palluzzi <fernando.palluzzi@gmail.com>

Examples

```
library(huge)
als.npn <- huge.npn(alsData$exprs)

adjdata <- SEMbap(alsData$graph, als.npn)$data

# Clusters creation
```

```
clusters <- extractClusters(graph = alsData$graph, data = adjdata)
head(parameterEstimates(clusters$fit$HM1))
head(parameterEstimates(clusters$fit$HM2))
head(parameterEstimates(clusters$fit$HM4))
gplot(clusters$clusters$HM2)

# Map cluster on the input graph
g <- alsData$graph
c <- clusters$clusters$HM2
V(g)$color <- ifelse(V(g)$name %in% V(c)$name, "gold", "white")
gplot(g)
```

---

gplot *Graph plotting with renderGraph*

Description

Wrapper for function renderGraph of the R package Rgraphwiz.

Usage

```
gplot(
  graph,
  l = "dot",
  main = "",
  cex.main = 1,
  font.main = 1,
  color.txt = "black",
  fontsize = 16,
  cex = 0.6,
  shape = "circle",
  color = "gray70",
  lty = 1,
  lwd = 1,
  w = "auto",
  h = "auto",
  psize = 80,
  ...
)
```

Arguments

graph	An igraph or graphNEL object.
l	any layout supported by `Rgraphviz`. It can be one among: "dot" (default), "neato", "circo", "fdp", "osage", "twopi".

main	Plot main title (by default, no title is added).
cex.main	Main title size (default = 1).
font.main	Main title font (default = 1). Available options are: 1 for plain text, 2 for bold, 3 for italics, 4 for bold italics, and 5 for symbol.
color.txt	Node text color (default = "black").
fontsize	Node text size (default = 16).
cex	Another argument to control node text size (default = 0.6).
shape	Node shape (default = "circle").
color	Node border color (default = "gray70").
lty	Node border outline (default = 1). Available options include: 0 for blank, 1 for solid line, 2 for dashed, 3 for dotted, 4 for dotdash, 5 for longdash, and 6 for twodash.
lwd	Node border thickness (default = 1).
w	Manual node width (default = "auto").
h	Manual node height (default = "auto").
psize	Automatic node size (default = 80).
...	Currently ignored.

Value

gplot returns invisibly the graph object produced by Rgraphviz

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
gplot(sachs$graph, main = "input graph")

sem <- SEMrun(sachs$graph, sachs$pkc)
gplot(sem$graph, main = "output graph")
```

---

graph2dag *Convert directed graphs to directed acyclic graphs (DAGs)*

Description

Remove cycles and bidirected edges from a directed graph.

Usage

```
graph2dag(graph, data, bap = FALSE, time.limit = Inf, ...)
```

Arguments

graph	A directed graph as an igraph object.
data	A data matrix with subjects as rows and variables as columns.
bap	If TRUE, a bow-free acyclic path (BAP) is returned (default = FALSE).
time.limit	CPU time for the computation, in seconds (default = Inf).
...	Currently ignored.

Details

The conversion is performed firstly by removing bidirected edges and then the data matrix is used to compute edge P-values, through marginal correlation testing (see weightGraph, r-to-z method). When a cycle is detected, the edge with highest P-value is removed, breaking the cycle. If the bap argument is TRUE, a BAP is generated merging the output DAG and the bidirected edges from the input graph.

Value

A DAG as an igraph object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
dag <- graph2dag(graph = sachs$graph, data = log(sachs$pkc))
old.par <- par(no.readonly = TRUE)
par(mfrow=c(1,2), mar=rep(1, 4))
gplot(sachs$graph, main = "Input graph")
gplot(dag, main = "Output DAG")
par(old.par)
```

---

graph2dagitty *Graph conversion from igraph to dagitty*

Description

Convert an igraph object to a dagitty object.

Usage

```
graph2dagitty(graph, canonical = FALSE, verbose = FALSE, ...)
```

Arguments

graph	A graph as an igraph or dagitty object.
canonical	A logical value. If TRUE, DAG conversion is enforced (for graph2dagitty only). This argument is FALSE by default.
verbose	A logical value. If TRUE, the output graph is shown (for graph2dagitty only). This argument is FALSE by default.
...	Currently ignored.

Value

A dagitty object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Graph as an igraph object to dagitty object
G <- graph2dagitty(sachs$graph)
plot(dagitty::graphLayout(G))
```

---

graph2lavaan *Graph to lavaan model*

Description

Convert an igraph object to a model (lavaan syntax).

Usage

```
graph2lavaan(graph, nodes = V(graph)$name, ...)
```

Arguments

graph	A graph as an igraph object.
nodes	Subset of nodes to be included in the model. By default, all the input graph nodes will be included in the output model.
...	Currently ignored.

Value

A model in lavaan syntax.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Graph (igraph object) to structural model in lavaan syntax
model <- graph2lavaan(sachs$graph)
cat(model, "\n")
```

---

kegg *KEGG interactome*

Description

Interactome generated by merging KEGG pathways extracted using the graphite R package (update: April, 2020).

Usage

```
kegg
```

Format

"kegg" is an igraph network object of 5934 nodes and 77158 edges corresponding to the union of 306 KEGG pathways.

Source

https://www.genome.jp/kegg/

References

Kanehisa M, Goto S (1999). KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acid Research 28(1): 27-30. <https://doi.org/10.1093/nar/27.1.29>

Martini P, Sales G, Massa MS, Chiogna M, Romualdi C. Along signal paths: anempirical gene set approach exploiting pathway topology. Nucleic Acids Res. 41(1):e19. <https://doi.org/10.1093/nar/gks866>

Examples

```
library(igraph)

# KEGG graph
summary(kegg)
```

```
# KEGG degrees of freedom
vcount(kegg)*(vcount(kegg) - 1)/2 - ecount(kegg)

# KEGG average shortest path length
mean_distance(kegg)
```

`lavaan2graph` *lavaan model to graph*

Description

Convert a model, specified using lavaan syntax, to a graph object in either igraph or dagitty format.

Usage

```
lavaan2graph(
  model,
  directed = TRUE,
  psi = TRUE,
  format = "igraph",
  verbose = FALSE,
  ...
)
```

Arguments

`model`	Model specified using lavaan syntax.
`directed`	Logical value. If TRUE (default), edge directions from the model will be preserved. If FALSE, the resulting graph will be undirected.
`psi`	Logical value. If TRUE (default) covariances will be converted into bidirected graph edges. If FALSE, covariances will be excluded from the output graph.
`format`	Output graph format. It can be either "igraph" (default) or "dagitty".
`verbose`	Logical value. If TRUE, a plot of the output graph will be generated. For large graphs, this could significantly increase computation time. If FALSE (default), graph plotting will be disabled.
`...`	Currently ignored.

Value

An igraph object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Writing path diagram in lavaan syntax

model<-'
#path model
Jnk ~ PKA + PKC
P38 ~ PKA + PKC
Akt ~ PKA + PIP3
Erk ~ PKA + Mek
Mek ~ PKA + PKC + Raf
Raf ~ PKA + PKC
PKC ~ PIP2 + Plcg
PIP2 ~ PIP3 + Plcg
Plcg ~ PIP3
#PKA ~ 1
#PIP3 ~ 1

# (co)variances
# PIP2 ~~ PIP3
'

# Graph with covariances
G0 <- lavaan2graph(model, psi = TRUE)
plot(G0, layout = layout.circle)

# Graph without covariances
G1 <- lavaan2graph(model, psi = FALSE)
plot(G1, layout = layout.circle)
```

`mergeNodes` *Graph nodes merging by a user-defined membership attribute*

Description

Merge groups of graph nodes using a custom membership attribute (e.g., cluster membership).

Usage

```
mergeNodes(graph, membership, HM, ...)
```

Arguments

`graph`	Network as an igraph object.
`membership`	Cluster membership. A vector of cluster membership identifiers, where vector names correspond to graph node names. Topological graph clustering can be done using `clusterGraph`.

HM	Hidden model label. If HM = "LV", a latent variable (LV) will be defined as common unknown cause acting on cluster nodes. If HM = "CV", cluster nodes will be considered as regressors of a latent composite variable (CV). Finally, if HM = "UV", an unmeasured variable (UV) is defined, where source nodes of the module (i.e., in-degree = 0) act as common regressors influencing the other nodes via an unmeasured variable.
...	Currently ignored.

Value

A network with merged nodes as an igraph object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

See Also

clusterGraph

Examples

```
# Clustering ALS graph with WTC method
G <- properties(alsData$graph)[[1]]
membership <- clusterGraph(graph = G, type = "wtc")
M <- mergeNodes(G, membership, HM = "LV")
```

---

modelSearch *Optimal model search strategies*

Description

Four model search strategies are implemented combining SEMdag(), SEMbap(), and extendGraph() functions. All strategies estimate a DAG through the adjusted (de-correlate) data matrix Z by iteratively update DAG and Z.

Usage

```
modelSearch(
  graph,
  data,
  gnet = NULL,
  d = 2,
  search = "basic",
  beta = 0,
  alpha = 0.05,
```

```
  pstop = TRUE,
  limit = 30000,
  verbose = FALSE,
  ...
)
```

Arguments

graph	Input network as an igraph object.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
gnet	Reference directed network used to validate and import nodes and interactions.
d	Maximum allowed geodesic distance for directed or undirected shortest path search. A distance d = 0 disables shortest path search (fixed in search = "basic"), while d = 1 (fixed in search = "direct") only search for directed links (i.e., no mediators are allowed). A distance d > 1 (defaults to d = 2 for "outer" and "inner" strategies), will search for shortest paths with at most d - 1 mediators between nodes sharing a significant estimated interaction. Connectors are imported from the reference interactome, as specified by the argument gnet. If the edges of the reference interactome are weighted by P-value, as defined by the E(graph)$pv attribute, the shortest path with the smallest sum of weights will be chosen (e.g., see weightGraph for graph weighting options).
search	Search strategy. Four model search strategies are available: • "outer". The estimated DAG is extended using extendGraph to find new indirect paths (i.e., inferred directed connections that may hide new mediators). New interactions and mediators will be searched and imported from the reference network (argument gnet, see above). Both DAG and extended graph complexity can be controlled with beta > 0 and d > 1 arguments, respectively (see below). The term "outer" means that new model mediator variables are imported from an external resource (i.e., the reference network). • "inner". This strategy is analogous to the "outer" one, but disables external mediator search. In other words, new indirect paths are generated by adding new interactions of the input model, so that mediators will be nodes already present in the input graph. The reference network is still used to validate new model paths. Also in this case, beta > 0 and d > 1 are used. • "direct". The input graph structure is improved through direct (i.e., adjacent) link search, followed by interaction validation and import from the reference network, with no mediators (i.e., d = 1). • "basic" (default). While the previous strategies rely on the input graph and the reference network to integrate knowledge to the final model, the "basic" strategy is data-driven. The input graph is needed to define the topological order. The argument gnet is set to NULL (i.e., no reference network is needed) and argument d = 0. Model complexity can be still controlled by setting beta > 0.

beta	Numeric value. Minimum absolute LASSO beta coefficient for a new interaction to be retained in the estimated DAG backbone. Lower beta values correspond to more complex DAGs. By default, beta is set to 0 (i.e., maximum complexity).
alpha	Significance level for false discovery rate (FDR) used for either local d-separation tests (below limit) or conditional independence (CI) test (above limit). This argument is used to control data de-correlation. A higher alpha level includes more hidden covariances, thus considering more sources of confounding. If alpha = 0, data de-correlation is disabled. By default, alpha = 0.05.
pstop	A logical value. With the argument pstop = TRUE (default), the algorithm can be halted when the Shipley's global model test P-value > 0.05. If pstop = FALSE, the model search algorithm stops when no additional edges can be added to the estimated DAG.
limit	An integer value corresponding to the number of missing edges of the extracted acyclic graph. Beyond this limit, multicore computation is enabled to reduce the computational burden.
verbose	If TRUE, it shows intermediate graphs during the execution (not recommended for large graphs).
...	Currently ignored.

Details

Search strategies can be ordered by decreasing conservativeness respect to the input graph, as: "direct", "inner", "outer", and "basic". The first three strategies are knowledge-based, since they require an input graph and a reference network, together with data, for knowledge-assisted model improvement. The last one does not require any reference and the output model structure will be completely determined by data. Output model complexity can be limited using arguments d and beta. While d is fixed to 0 or 1 in "basic" or "direct", respectively; we suggest starting with d = 2 (only one mediator) for the other two strategies. For knowledge-based strategies, we suggest to to start with beta = 0.1. Then, beta can be relaxed (0 to < 0.1) to improve model fitting, if needed. Since data-driven models can be complex, we suggest to start from beta = 0.1 when using the "basic" strategy. The beta value can be relaxed until a good model fit is obtained. Argument alpha determines the extent of data adjustment: lower alpha values for FDR correction correspond to a smaller number of significant confounding factors, hence a weaker correction (default alpha = 0.05).

Value

The output model as well as the adjusted dataset are returned as a list of 3 objects:

- "fit", the fitted output model (lavaan object);
- "graph", the output model as an igraph object;
- "data", the adjusted dataset.

Author(s)

Fernando Palluzzi <fernando.palluzzi@gmail.com>

Examples

```
# Comparison among different model estimation strategies

library(huge)
als.npn <- huge.npn(alsData$exprs)

# Models estimation
m1 <- modelSearch(graph = alsData$graph, data = als.npn, gnet = kegg,
      search = "direct", beta = 0, alpha = 0.05)
m2 <- modelSearch(graph = alsData$graph, data = als.npn, gnet = kegg,
      d = 2, search = "inner", beta = 0.05, alpha = 0.05)
m3 <- modelSearch(graph = alsData$graph, data = als.npn, gnet = kegg,
      d = 2, search = "outer", beta = 0.05, alpha = 0.05)
m4 <- modelSearch(graph = alsData$graph, data = als.npn, gnet = NULL,
      search = "basic", beta = 0.1, alpha = 0.05)

# Graphs
#old.par <- par(no.readonly = TRUE)
#par(mfrow=c(2,2), mar= rep(1,4))
gplot(m1$graph, main = "direct graph")
gplot(m2$graph, main = "inner graph")
gplot(m3$graph, main = "outer graph")
gplot(m4$graph, main = "basic graph")
#par(old.par)
```

`orientEdges` *Assign edge orientation of an undirected graph*

Description

Assign edge orientation of an undirected graph through a given reference directed graph.

Usage

```
orientEdges(ug, dg, ...)
```

Arguments

`ug`	An undirected graph as an igraph object.
`dg`	A directed reference graph.
`...`	Currently ignored.

Value

A directed graph as an igraph object.

Examples

```
# Graphs definition
G0 <- corr2graph(R = cor(log(sachs$pkc)), n = nrow(sachs$pkc), type = "marg")

# Reference graph-based orientation
G1 <- orientEdges(ug = G0, dg = sachs$graph)

# Graphs plotting
old.par <- par(no.readonly = TRUE)
par(mfrow=c(1,2), mar=rep(2,4))
plot(G0, layout=layout.circle, main = "Input undirected graph")
plot(G1, layout=layout.circle, main = "Output directed graph")
par(old.par)
```

---

pairwiseMatrix *Pairwise plotting of multivariate data*

Description

Display a pairwise scatter plot of two datasets for a random selection of variables. If the second dataset is not given, the function displays a histogram with normal curve superposition.

Usage

```
pairwiseMatrix(x, y = NULL, size = nrow(x), r = 4, c = 4, ...)
```

Arguments

x	A matrix or data.frame (n x p) of continuous data.
y	A matrix or data.frame (n x q) of continuous data.
size	number of rows to be sampled (default s = nrow(z)).
r	number of rows of the plot layout (default r = 4).
c	number of columns of the plot layout (default r = 4).
...	Currently ignored.

Value

No return value

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
adjdata <- SEMbap(sachs$graph, log(sachs$pkc))$data
rawdata <- log(sachs$pkc)
pairwiseMatrix(adjdata, rawdata, size = 1000)
```

`pathFinder` *Perturbed path search utility*

Description

This function uses `SEMace` to find significant causal effects between source-sink pairs and `SEMpath` to fit them and test their edge perturbation.

Usage

```
pathFinder(
  graph,
  data,
  group = NULL,
  ace = NULL,
  path = "directed",
  method = "none",
  alpha = 0.05,
  verbose = FALSE,
  ...
)
```

Arguments

`graph`	Input network as an igraph object.
`data`	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
`group`	group A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. Group specification enables edge perturbation testing. By default, `group = NULL`.
`ace`	A data.frame generated by `SEMace`. If NULL, `SEMace` will be automatically run.
`path`	If `path = "directed"`, all directed paths between the two nodes will be included in the fitted model. If `path = "shortest"`, only shortest paths will be considered.
`method`	Multiple testing correction method. One of the values available in `p.adjust`. By default, `method = "none"` (i.e., no multiple test correction).
`alpha`	Significance level for ACE selection (by default, `alpha = 0.05`).
`verbose`	Show the significant directed (or shortest) paths inside the input graph.
`...`	Currently ignored.

Value

A list of 3 objects:

- "paths", list of paths as igraph objects;
- "fit", fitting results for each path as a lavaan object;
- "dfp", a data.frame containing SEM global fitting statistics.

Author(s)

Fernando Palluzzi <fernando.palluzzi@gmail.com>

Examples

```
# Find and evaluate significantly perturbed paths

library(huge)
als.npn <- huge.npn(alsData$exprs)

adjData <- SEMbap(graph = alsData$graph, data = als.npn)$data

ace <- SEMace(graph = alsData$graph, data = adjData,
              group = alsData$group)
ace <- ace[order(ace$pvalue),]
print(ace)

paths <- pathFinder(graph = alsData$graph, data = adjData,
                    group = alsData$group,
                    ace = ace)

head(parameterEstimates(paths$fit$P19))
gplot(paths$paths$P19)

path19 <- SEMpath(graph = alsData$graph, data = adjData,
                  group = alsData$group,
                  from = "7133",
                  to = "4747",
                  path = "directed",
                  verbose = TRUE)
```

properties *Graph properties summary and graph decomposition*

Description

Produces a summary of network properties and returns graph components (ordered by decreasing size), without self-loops.

Usage

```
properties(graph, data = NULL, ...)
```

Arguments

graph	Input network as an igraph object.
data	An optional data matrix whith rows corresponding to subjects, and columns to graph nodes (variables). Nodes will be mapped onto variable names.
...	Currently ignored.

Value

List of graph components, ordered by decreasing size (the first component is the giant one), without self-loops.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Extract from graphite the "Steroid biosynthesis" pathway:

library(graphite)
humanKegg <- pathways("hsapiens", "kegg")
p <- humanKegg[["Steroid biosynthesis"]]
g <- pathwayGraph(p)
graph::nodes(g) <- gsub("ENTREZID:","",graph::nodes(g))

properties(graph_from_graphnel(g))
```

sachs *Sachs multiparameter flow cytometry data and consensus model*

Description

Flow cytometry data and causal model from Sachs et al. (2005).

Usage

```
sachs
```

Format

"sachs" is a list of 5 objects:

1. "rawdata", a list of 14 data.frames containing raw flow cytometry data (Sachs et al., 2005);
2. "graph", consensus signaling network;
3. "model", consensus model (lavaan syntax);
4. "pkc", data.frame of 1766 samples and 11 variables, containing cd3cd28 (baseline) and pma (PKC activation) data;
5. "group", a binary group vector, where 0 is for cd3cd28 samples (n = 853) and 1 is for pma samples (n = 913).
6. "details", a data.frame containing dataset information.

Source

doi: 10.1126/science.1105809

References

Sachs K, Perez O, Pe'er D, Lauffenburger DA, Nolan GP (2019). Causal Protein-Signaling Networks Derived from Multiparameter Single-Cell Data. Science, 308(5721): 523-529.

Examples

```
# Dataset content
names(sachs$rawdata)
dim(sachs$pkc)
table(sachs$group)
cat(sachs$model)
gplot(sachs$graph)
```

SEMace *Compute the Average Causal Effect (ACE) for a given source-sink pair*

Description

Compute total effects as ACEs of source variables X (i.e., incoming connectivity = 0) on sink variables Y (i.e., outgoing connectivity = 0), in a directed graph. The ACE will be estimated as the path coefficient of X (i.e., theta) in the linear equation Y ~ X + Z. Z is defined as the adjustment (or conditioning) set of Y over X, applying an "optimal" valid set (O-set), with the smallest asymptotic variance. Standard errors (SE), for each ACE, are computed following the `lavaan` standard procedure or a bootstrap-based procedure (see `boot` for details).

Usage

```
SEMace(
  graph,
  data,
  group = NULL,
  method = "none",
  alpha = 0.05,
  boot = NULL,
  ...
)
```

Arguments

`graph`	An igraph object.
`data`	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
`group`	A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. If `group = NULL` (default), group influence will not be considered.
`method`	Multiple testing correction method. One of the values available in `p.adjust`. By default, `method = "none"` (i.e., no multiple test correction).
`alpha`	Significance level for ACE selection (by default, `alpha = 0.05`).
`boot`	The number of bootstrap samplings enabling bootstrap computation of ACE standard errors. If `NULL` (default), bootstrap is disabled.
`...`	Currently ignored.

Value

A data.frame of ACE estimates between network sources and sinks.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Witte J, Henckel L, Maathuis MH, Didelez V (2020). On efficient adjustment in causal graphs. arXiv:2002.06825 [math.ST]. <https://arxiv.org/abs/2002.06825/>

Examples

```
# ACE estimation, without group (default)
ace <- SEMace(graph = sachs$graph, data = log(sachs$pkc))
print(ace)

# ACE estimation, with group perturbation and multiple test correction
ace2 <- SEMace(graph = sachs$graph, data = log(sachs$pkc),
               group = sachs$group,
               method = "BH", alpha = 0.05)
print(ace2)
```

SEMbap *Bow-free covariance search and data de-correlation*

Description

Search for new bow-free covariances and adjust the data matrix by removing latent sources of confounding encoded in them.

Usage

```
SEMbap(
  graph,
  data,
  method = "BH",
  alpha = 0.05,
  limit = 30000,
  verbose = FALSE,
  ...
)
```

Arguments

graph	An igraph object.
data	A matrix whith rows corresponding to subjects, and columns to graph nodes (variables).
method	Multiple testing correction method. One of the values available in p.adjust. By default, method is set to "BH" (i.e., Benjamini-Hochberg multiple test correction).

alpha	Significance level for false discovery rate (FDR) used for either local d-separation tests (below limit) or conditional independence (CI) test (above limit). This argument is used to control data de-correlation. A higher alpha level includes more hidden covariances, thus considering more sources of confounding. If alpha = 0, data de-correlation is disabled. By default, alpha = 0.05.
limit	An integer value corresponding to the number of missing edges of the extracted acyclic graph. Beyond this limit, multicore computation is enabled to reduce the computational burden. By default, limit = 30000.
verbose	A logical value. If FALSE (default), the processed graphs will not be plotted to screen.
...	Currently ignored.

Details

SEMbap algorithm makes an exhaustive search of all possible missing edges of the mixed acyclic graph (BAP or DAG) via d-separation P-value screening. The d-separation test evaluates if two variables (X, Y) in an acyclic graph are conditionally independent for a given conditioning set Z, The conditioning set Z is represented in a DAG by the union of the parent sets of X and Y (Shipley, 2000) or the minimal set consisting in the smallest conditioning set Z that makes these two variables independent. A new bow-free covariance is added if there is a significant (X, Y) association, after multiple testing correction. The selected covariance between pairs of nodes (X, Y) is interpreted as the effect of a latent variable (LV) acting on both X and Y; i.e., the LV is an unobserved confounder. These LVs are then removed by conditioning them out from the observed data.

Value

A list of 3 igraph objects:

- "bap", the output bow-free acyclic path diagram,
- "guu", the bidirected graph of significant covariances,
- "gLV", the directed graph of latent variables (LV) underlying significant covariances (i.e., the canonical graph, where bidirected X <-> Y edges are substituted by directed edges X <- LV -> Y),
- "data", the adjusted (de-correlated) data matrix.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Shipley B (2000). A new inferential test for path models based on DAGs. Struct. Equ. Modeling, 7(2): 206-218. <https://doi.org/10.1207/S15328007SEM0702_4>

Brito C and Pearl J (2002). A New Identification Condition for Recursive Models With Correlated Errors. Structural Equation Modeling, 9(4): 459-474.

Whittaker J (2009). Graphical Models in Applied Multivariate Statistics. Wiley Publishing; <ISBN:978-0-470-74366-9>

Examples

```
# Model fitting
sem0 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc))

# BAP estimation
BAP <- SEMbap(graph = sachs$graph, data = log(sachs$pkc), verbose = TRUE)

# Model fitting (node perturbation) with adjusted data
sem1 <- SEMrun(graph = sachs$graph, data = BAP$data, group = sachs$group)
```

SEMdag *Estimate the optimal DAG from an input graph*

Description

Extract the optimal DAG from an input graph, using the LASSO-based algorithm, implememted in glmnet.

Usage

```
SEMdag(
  graph,
  data,
  gnet = NULL,
  d = 0,
  beta = 0,
  lambdas = NA,
  verbose = FALSE,
  ...
)
```

Arguments

graph	An igraph object.
data	A matrix whith rows corresponding to subjects, and columns to graph nodes (variables).
gnet	Reference "global" network as an igraph object. If given, new edges will be added to the final DAG only if present in the reference network.
d	An integer value indicating the maximum length of indirect interactions between pairs of nodes. If d = 1, direct interactions between nodes will be searched in the reference interactome (if given). If d > 1, indirect interactions of length d or shorter (i.e., with at most d -1 connectors) between bow-free nodes will be searched. Setting d = 0, is equivalent to gnet = NULL.
beta	Numeric value. Minimum absolute LASSO beta coefficient for a new interaction to be retained in the final model. By default, beta = 0.

lambdas	A vector of regularization LASSO lambda values. Cross-validation (n > 100) or BIC-based (n <= 100) optimal lambdas for each response variable will be selected. If lambdas is NULL, the glmnet default is enabled. If lambdas is NA (default), the tuning-free scheme is enabled by fixing lambdas = sqrt(log(p)/n), as suggested by Janková and van de Geer (2015). This will both reduce computational time and provide the same result at each run.
verbose	A logical value. If FALSE (default), the processed graphs will not be plotted to screen.
...	Currently ignored.

Details

The optimal DAG is estimated after node topological order, using successive penalized (L1) regressions. If the input graph is not acyclic, a warning message will be raised, and a cycle-breaking algorithm will be applied (see graph2dag for details). Output DAG edges will be colored in blue, if they were present in the input graph, and in red, if they are new edges generated by LASSO screening.

Value

A list of 3 igraph objects:

1. "dag", the estimated DAG;
2. "dag.red", new estimated connections;
3. "dag.blue", connections preserved from the input graph.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Shojaie A, Michailidis G (2010). Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. Biometrika, 97(3): 519-538. <https://doi.org/10.1093/biomet/asq038>

Tibshirani R, Bien J, Friedman J, Hastie T, Simon N, Taylor J, Tibshirani RJ (2012). Strong rules for discarding predictors in lasso type problems. Royal Statistical Society: Series B (Statistical Methodology), 74(2): 245-266. <https://doi.org/10.1111/j.1467-9868.2011.01004.x>

Jana Jankova and Sara van de Geer (2015). Confidence intervals for high-dimensional inverse covariance estimation. Electronic Journal of Statistics, 9(1): 1205-1229. <https://doi.org/10.1214/15-EJS1031>

See Also

modelSearch

Examples

```
# DAG estimation
G <- SEMdag(graph = sachs$graph, data = log(sachs$pkc), beta = 0.05)

# Model fitting
sem <- SEMrun(graph = G$dag, data = log(sachs$pkc), group = sachs$group)

# Graphs
old.par <- par(no.readonly = TRUE)
par(mfrow=c(2,2), mar=rep(1,4))
plot(sachs$graph, layout=layout.circle, main="input graph")
plot(G$dag, layout=layout.circle, main = "Output DAG")
plot(G$dag.blue, layout=layout.circle, main = "Inferred old edges")
plot(G$dag.red, layout=layout.circle, main = "Inferred new edges")
par(old.par)
```

SEMgsa *SEM-based gene set analysis*

Description

Gene Set Analysis (GSA) via self-contained test for group effect on signaling (directed) pathways as SEM, evaluating overall pathway perturbation, perturbation emission from source nodes, and perturbation accumulation on target nodes. Approximate randomization test P-values of specific node and aggregated group effects will be computed. For directed graphs, they include: the sum of group effects adjusted by residual variances (D), the sum of the tagret nodes perturbation (i.e., group effect) accumulation from source nodes (A), and the sum of the source nodes perturbation emission towards target nodes (E). For undirected graphs, the sum of group effects, adjusted by residual variances (D), will be estimated.

Usage

```
SEMgsa(g = list(), data, group, method = "BH", alpha = 0.05, n_rep = 1000, ...)
```

Arguments

g	A list of pathways to be tested.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
group	A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects.
method	Multiple testing correction method. One of the values available in p.adjust. By default, method is set to "BH" (i.e., Benjamini-Hochberg correction).
alpha	Gene set test significance level. Alpha is set to 0.05 by default.
n_rep	Number of randomization replicates (default = 1000).
...	Currently ignored.

Value

A list of 2 objects:

1. "gsa", A data.frame reporting the following information for each pathway in the input list:
   - "N.nodes", pathway size (number of nodes);
   - "N.DRNs", number of differential regulated genes (DRNs) within the pathway, after multiple test correction with Benjamini-Hochberg method;
   - "pD", significance of the sum of group effects, adjusted by the residual variance;
   - "pA", significance of the sum of tagret nodes perturbation (i.e., group effect) accumulation from source nodes;
   - "pE", significance of the sum of source nodes perturbation (i.e., group effect) emission towards target nodes;
   - "pvalue", Fisher's combined P-value of pD, pA, and pE.
2. "DRN", a list with DRNs names per pathways.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
## Not run:

# Install data examples, reference networks, and pathways:
# devtools::install_github("fernandoPalluzzi/SEMdata")
library(SEMdata)
library(huge)

als.npn <- huge.npn(alsData$exprs)

#selection of FTD pathways from KEGG

paths.name <- c("MAPK signaling pathway",
                "Protein processing in endoplasmic reticulum",
                "Endocytosis",
                "Wnt signaling pathway",
                "Notch signaling pathway",
                "Neurotrophin signaling pathway",
                "Amyotrophic lateral sclerosis (ALS)")

j <- which(names(kegg.pathways) %in% paths.name)
GSA <- SEMgsa(kegg.pathways[j], als.npn, alsData$group,
              method = "bonferroni", alpha = 0.05,
              n_rep = 1000)
GSA$gsa
GSA$DRN


## End(Not run)
```

SEMpath *Search for directed or shortest paths between pairs of source-sink nodes*

Description

Find and fit all directed or shortest paths between two source-sink nodes of a graph.

Usage

```
SEMpath(graph, data, group, from, to, path, verbose = FALSE, ...)
```

Arguments

graph	An igraph object.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes (variables).
group	A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. If NULL (default), group influence will not be considered.
from	Starting node name (i.e., source node).
to	Ending node name (i.e., sink node).
path	If path = "directed", all directed paths between the two nodes will be included in the fitted model. If path = "shortest", only shortest paths will be returned.
verbose	Show the directed (or shortest) path between the given source-sink pair inside the input graph.
...	Currently ignored.

Value

A list of four objects: a fitted model object of class lavaan ("fit"), aggregated and node-specific group effect estimates and P-values ("gest"), the extracted subnetwork as an igraph object ("graph"), and the input graph with a color attribute mapping the chosen path ("map").

Author(s)

Mario Grassi <mario.grassi@unipv.it>

Examples

```
# Directed path fitting
path <- SEMpath(graph = sachs$graph, data = log(sachs$pkc),
                group = sachs$group,
                from = "PIP3",
                to = "Erk",
                path = "directed")

# Summaries
summary(path$fit)
print(path$gest)

# Graphs
gplot(path$map, main="path from PiP2 to Erk")
plot(path$map, layout=layout.circle, main="path from PiP2 to Erk")
```

SEMrun *Fit a graph as a Structural Equation Model (SEM)*

Description

SEMrun() converts a (directed, undirected, or mixed) graph to a SEM and fits it. If a binary group variable (i.e., case/control) is present, node-level or edge-level perturbation is evaluated. This function can handle loop-containing models, although multiple links between the same two nodes (including self-loops and mutual interactions) and bows (i.e., a directed and a bidirected link between two nodes) are not allowed.

Usage

```
SEMrun(
  graph,
  data,
  group = NULL,
  fit = 0,
  algo = "lavaan",
  start = NULL,
  limit = 100,
  ...
)
```

Arguments

graph	An igraph object.
data	A matrix whith rows corresponding to subjects, and columns to graph nodes (variables).

`group` A binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. If `NULL` (default), group influence will not be considered.

`fit` A numeric value indicating the SEM fitting mode. If `fit = 0` (default), no group effect is considered. If `fit = 1`, a "common" model is used to evaluate group effects on graph nodes. If `fit = 2`, a two-group model is used to evaluate group effects on graph edges.

`algo` MLE method used for SEM fitting. If `algo = "lavaan"` (default), the SEM will be fitted using the NLMINB solver from `lavaan` R package, with standard errors derived from the expected Fisher information matrix. If `algo = "ricf"`, the model is fitted via residual iterative conditional fitting (RICF; Drton et al. 2009). If `algo = "cggm"`, model fitting is based on constrained Gaussian Graphical Modeling (GGM) and de-sparsified glasso estimator (Williams, 2020).

`start` Starting value of SEM parameters for `algo = "lavaan"`. If start is `NULL` (default), the algorithm will determine the starting values. If start is a numeric value, it will be used as a scaling factor for the edge weights in the graph object (graph attribute `E(graph)$weight`). For instance, a scaling factor is useful when weights have fixed values (e.g., 1 for activated, -1 for repressed, and 0 for unchanged interaction). Fixed values may compromise model fitting, and scaling them is a safe option to avoid this problem. As a rule of thumb, to our experience, `start = 0.1` generally performs well with -1, 0, 1 weights.

`limit` An integer value corresponding to the network size (i.e., number of nodes). Beyond this limit, the execution under `algo = "lavaan"` will be ridirected to `algo = "ricf"`, if fit is either 0 or 1, or to `algo = "ggm"`, if `fit = 2`. This redirection is necessary to reduce the computational demand of standard error estimation by lavaan. Increasing this number will enforce lavaan execution when `algo = "lavaan"`.

`...` Currently ignored.

Details

SEMrun maps data onto the input graph and converts it into a SEM. Directed connections (X -> Y) are interpreted as direct causal effects, while undirected, mutual, and bidirected connections are converted into model covariances. SEMrun output contains different sets of parameter estimates. Beta coefficients (i.e., direct effects) are estimated from directed interactions and residual covariances (psi coefficients) from bidirected, undirected, or mutual interactions. If a group variable is given, exogenous group effects on nodes (gamma coefficients) will be estimated. This will also lead to the estimation of a set of aggregated group effects, if `algo = "ricf"` (see `SEMgsa`). By default, maximum likelihood parameter estimates and P-values for parameter sets are computed by conventional z-test (= estimate/SE), and fits it through the `lavaan` function, via Maximum Likelihood Estimation (estimator = "ML", default estimator in `lavOptions`). In case of high dimensionality (n.variables » n.subjects), the covariance matrix could not be semi-definite positive and thus parameter estimates could not be done. If this happens, covariance matrix regularization is enabled using the James-Stein-type shrinkage estimator implemented in the function `pcor.shrink` of corpcor R package. Argument `fit` determines how group influence is evaluated in the model, as absent (`fit = 0`), node perturbation (`fit = 1`), or edge perturbation (`fit = 2`). When `fit = 1`, the group is modeled as an exogenous variable, influencing all the other graph nodes. When `fit = 2`, SEMrun estimates

the differences of the beta and/or psi coefficients (network edges) between groups. This is equivalent to fit a separate model for cases and controls, as opposed to one common model perturbed by the exogenous group effect. Once fitted, the two models are then compared to assess significant edge (i.e., direct effect) differences (d = beta1 - beta0). P-values for parameter sets are computed by z-test (= d/SE), through `lavaan`. As an alternative to standard P-value calculation, SEMrun may use either RICF (randomization P-values) or GGM (de-sparsified P-values) methods. These algorithms are much faster than `lavaan` in case of large input graphs.

Value

A list of 5 objects:

1. "fit", SEM fitted lavaan, ricf, or ggmncv object, depending on the MLE method specified by the `algo` argument;
2. "gest" or "dest", a data.frame of node-specific ("gest") or edge-specific ("dest") group effect estimates and P-values;
3. "model", SEM model as a string if `algo = "lavaan"`, and `NULL` otherwise;
4. "graph", the induced subgraph of the input network mapped on data variables. Graph edges (i.e., direct effects) with P-value < 0.05 will be highlighted in red (beta > 0) or blue (beta < 0). If a group vector is given, nodes with significant group effect (P-value < 0.05) will be red-shaded (beta > 0) or lightblue-shaded (beta < 0);
5. "dataXY", input data subset mapping graph nodes, plus group at the first column (if no group is specified, this column will take NA values).

Author(s)

Mario Grassi `<mario.grassi@unipv.it>`

References

Pearl J (1998). Graphs, Causality, and Structural Equation Models. Sociological Methods & Research., 27(2):226-284. <https://doi.org/10.1177/0049124198027002004>

Yves Rosseel (2012). lavaan: An R Package for Structural Equation Modeling. Journal of Statistical Software, 48(2): 1-36. <https://www.jstatsoft.org/v48/i02/>

Pepe D, Grassi M (2014). Investigating perturbed pathway modules from gene expression data via Structural Equation Models. BMC Bioinformatics, 15: 132. <https://doi.org/10.1186/1471-2105-15-132>

Drton M, Eichler M, Richardson TS (2009). Computing Maximum Likelihood Estimated in Recursive Linear Models with Correlated Errors. Journal of Machine Learning Research, 10(Oct): 2329-2348. <https://www.jmlr.org/papers/volume10/drton09a/drton09a.pdf>

Larson JL and Owen AB (2015). Moment based gene set tests. BMC Bioinformatics, 16: 132. <https://doi.org/10.1186/s12859-015-0571-7>

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

Williams D (2020). GGMncv: Gaussian Graphical Models with Non-Convex Penalties. R package version 1.1.0. <https://CRAN.R-project.org/package=GGMncv/>

See Also

See fitAncestralGraph for RICF algorithm details, flip for randomization P-values, and constrained for constrained GGM, and inference for de-sparsified P-values.

Examples

```
#### Model fitting (no group effect)

sem0 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc), algo = "lavaan")
summary(sem0$fit)
head(parameterEstimates(sem0$fit))

sem0 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc), algo = "ricf")
summary(sem0$fit)
head(sem0$fit$parameterEstimates)

sem0 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc), algo = "cggm")
summary(sem0$fit)
head(sem0$fit$parameterEstimates)

# Graphs
gplot(sem0$graph, main = "edge differences")
plot(sem0$graph, layout = layout.circle, main = "edge differences")


#### Model fitting (common model, group effect on nodes)

sem1 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc),
               group = sachs$group)

# Fitting summaries
summary(sem1$fit)
print(sem1$gest)
head(parameterEstimates(sem1$fit))

# Graphs
gplot(sem1$graph, main = "node differences")
plot(sem1$graph, layout = layout.circle, main = "node differences")


#### Two-group model fitting (group effect on edges)

sem2 <- SEMrun(graph = sachs$graph, data = log(sachs$pkc),
               group = sachs$group,
               fit = 2)

# Summaries
summary(sem2$fit)
print(sem2$dest)
head(parameterEstimates(sem2$fit))
```

```
# Graphs
gplot(sem2$graph, main = "Between group edge differences")
plot(sem2$graph, layout = layout.circle, main = "Between group edge differences")



# Fitting and visualization of a large pathway:

library(graphite)
humanKegg <- pathways("hsapiens", "kegg")
p <- humanKegg[["MAPK signaling pathway"]]
g <- pathwayGraph(p)
graph::nodes(g) <- gsub("ENTREZID:","",graph::nodes(g))
G <- properties(graph_from_graphnel(g))[[1]]

library(huge)
als.npn <- huge.npn(alsData$exprs)

g1 <- SEMrun(G, als.npn, alsData$group, algo = "cggm")$graph
g2 <- SEMrun(g1, als.npn, alsData$group, fit = 2, algo = "cggm")$graph

# extract the subgraph with between group node and edge differences
g2 <- g2 - E(g2)[-which(E(g2)$color != "gray50")]
g <- properties(g2)[[1]]

# plot graph
library(org.Hs.eg.db)
V(g)$label <- mapIds(org.Hs.eg.db, V(g)$name, 'SYMBOL', 'ENTREZID')
E(g)$color<- E(g2)$color[E(g2) %in% E(g)]
gplot(g, l = "fdp", main="node and edge group differences")
```

`Shipley.test` *Missing edge testing implied by a graph*

Description

Compute all the P-values of the d-separation tests implied by the missing edges of a given acyclic graph (DAG or BAP). The conditioning set Z is represented, in a DAG, by the union of the parent sets of X and Y (Shipley, 2000). In a BAP, Z is the minimal set consisting in the smallest conditioning set Z that makes these two variables independent. The results of every test, in a DAG, is then combined using the Fisher's statistic in an overall test of the fitted model C = -2*sum(log(P-value(k))), where C is distributed as a chi-squared variate with df = 2k, as suggested by Shipley (2000). In a BAP, the P-values resulting from every test are corrected by multiple testing multiplying by the number of missing edges. The smallest one is then considered as the overall test P-value (Shipley, 2002).

Usage

```
Shipley.test(graph, data, limit = 30000, verbose = TRUE, ...)
```

Arguments

graph	A directed graph as an igraph object.
data	A data matrix with subjects as rows and variables as columns.
limit	An integer value corresponding to the number of missing edges of the extracted acyclic graph. Beyond this limit, multicore computation is enabled to reduce the computational burden. By default, `limit = 30000`.
verbose	If TRUE, Shipley's test results will be showed to screen (default = TRUE).
...	Currently ignored.

Value

A list of three objects: (i) the list of all d-separation tests over missing edges in the input DAG or BAP, (ii) the DAG or BAP used to perform the Shipley test, and (iii) the overall Shipley's P-value.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Shipley B (2000). A new inferential test for path models based on DAGs. Struct. Equ. Modeling, 7(2): 206-218. <https://doi.org/10.1207/S15328007SEM0702_4>

Shipley B (2002). Start and Stop Rules for Exploratory Path Analysis. Structural Equation Modeling A Multidisciplinary Journal, 9(4): 554-561. <https://doi.org/10.1207/S15328007SEM0904_5>

Examples

```
#\donttest{

library(huge)
als.npn <- huge.npn(alsData$exprs)

sem <- SEMrun(alsData$graph, als.npn)
C.test <- Shipley.test(sem$graph, als.npn)

#}
```

summary.GGM *GGM model summary*

Description

Generate a summary for a constrained Gaussian Graphical Model (GGM) similar to lavaan-formated summary

Usage

```
## S3 method for class 'GGM'
summary(object, ...)
```

Arguments

object	A constrained GGM fitted model object.
...	Currently ignored.

Value

Shown the lavaan-formatted summary to console

Author(s)

Mario Grassi <mario.grassi@unipv.it>

See Also

SEMrun.

Examples

```
sem0 <- SEMrun(sachs$graph, log(sachs$pkc), algo = "cggm")
summary(sem0$fit)
```

summary.RICF *RICF model summary*

Description

Generate a summary for a RICF solver similar to lavaan-formatted summary

Usage

```
## S3 method for class 'RICF'
summary(object, ...)
```

Arguments

object	A RICF fitted model object.
...	Currently ignored.

Value

Shown the lavaan-formatted summary to console

Author(s)

Mario Grassi <mario.grassi@unipv.it>

See Also

SEMrun.

Examples

```
sem0 <- SEMrun(sachs$graph, log(sachs$pkc), algo = "ricf")
summary(sem0$fit)
```

---

weightGraph *Graph weighting methods*

Description

Add data-driven edge and node weights to the input graph.

Usage

```
weightGraph(
  graph,
  data,
  group = NULL,
  method = "r2z",
  seed = "none",
  limit = 3000,
  ...
)
```

Arguments

graph	An igraph object.
data	A matrix or data.frame. Rows correspond to subjects, and columns to graph nodes.
group	Binary vector. This vector must be as long as the number of subjects. Each vector element must be 1 for cases and 0 for control subjects. By default, group = NULL.
method	Edge weighting method. It can be one of the following: 1. "r2z", Weight edges of a graph using Fisher's r-to-z transform to test the group difference between correlation coefficients of pairs of interacting nodes (Fisher, 1915). 2. "sem". Edge weights are defined by a SEM model that implies testing the group effect simultaneously on the j-th source node and the k-th sink node. A new parameter w is defined as the weighted sum of the total effect of the group on source and sink nodes, adjusted by node degree centrality, and edge weights correspond to the sign and P-value of the z-test = w/SE(w). Not available if group == NULL. 3. "cov". Edge weights are defined by a new parameter w combining the group effect on the source node (mean group difference, adjusted by source degree centrality), the sink node (mean group difference, adjusted by sink degree centrality), and the source-sink interaction (correlation difference). Edge weights correspond to the sign and P-value of the z-test = w/SE(w) of the combined difference of the group over source node, sink node, and their connection. Not available if group == NULL.
seed	A vector of three cutoffs. By default, seed = "none" and seed calculation is disabled. Suggested cutoff values are seed = c(0.05,0.5,0.5). If these cutoffs are defined, seed search is enabled. Nodes can be labeled as either seeds (node weight = 1) or non-seeds (node weight = 0), according to three alternative importance criteria: perturbed group effect, prototype clustering, and closeness node index. The first cutoff is the significance level of the group effect over graph nodes. The second is a threshold corresponding to the prototype clustering distance measure (= 1 - abs(correlation)) cutoff. The third one is the closeness percentile. Nodes having closeness greater than the q-th percentile are labeled as seeds. If the seed argument is enabled, the output graph will have three new binary (1: seed, 0: non-seed) vertex attibutes: 1. "pvlm", P-value of the simple linear regression y ~ x (i.e., node ~ group); 2. "proto", prototype seeds derived from protoclust; 3. "qi", nodes with closeness greater than the q-th percentile.
limit	An integer value corresponding to the number of graph edges. Beyond this limit, multicore computation is enabled to reduce the computational burden. By default, limit = 3000.
...	Currently ignored.

Value

A weighted graph, as an igraph object.

Author(s)

Mario Grassi <mario.grassi@unipv.it>

References

Palluzzi F, Grassi M (2021). SEMgraph: An R Package for Causal Network Analysis of High-Throughput Data with Structural Equation Models. <arXiv:2103.08332>

Examples

```
# Graph weighting
G <- weightGraph(graph = sachs$graph,
                 data = log(sachs$pkc),
                 group = sachs$group,
                 method = "r2z",
                 seed = c(0.05, 0.5, 0.5))

# New edge attributes
E(G)$pv
E(G)$zsign

# New nodes attributes (1: seed, 0: non-seed)
V(G)$pvlm; table(V(G)$pvlm)
V(G)$proto; table(V(G)$proto)
V(G)$qi; table(V(G)$qi)

# Reduced graph (using highest closeness nodes)
R <- induced_subgraph(G, vids = V(G)$name[V(G)$qi == 1])
R <- properties(R)[[1]]
```

Index

activeModule, 2
alsData, 5
ancestors *(*ancestry*)*, 6
ancestry, 6

boot, *34*

clusterGraph, 7, *9, 10*, *14*, *17*, *24, 25*
clusterScore, *7, 8*, 8, *14*
colorGraph, 10
constrained, *45*
corr2graph, 12
cplot, *8*, *10*, 13

descendants *(*ancestry*)*, 6

extendGraph, 14, *26*
extractClusters, 16

factor.analysis, *10*
fitAncestralGraph, *45*
flip, *45*

glmnet, *37, 38*
gplot, 18
graph2dag, 19, *38*
graph2dagitty, 20
graph2lavaan, 21

igraph, *4*, *14*
inference, *45*

kegg, 22

lavaan, *41*, *43, 44*
lavaan2graph, 23
lavOptions, *43*

mergeNodes, *9*, 24
modelSearch, 25, *38*

orientEdges, 28

p.adjust, *11, 12*, *30*, *34, 35*, *39*
pairwiseMatrix, 29
parents *(*ancestry*)*, 6
pathFinder, 30
pcor.shrink, *43*
properties, 32
protoclust, *50*

sachs, 33
SEMace, *30*, 34
SEMbap, *15*, 35
SEMdag, *15*, 37
SEMgsa, 39, *43*
SEMpath, *30*, 41
SEMrun, *10*, 42, *48, 49*
Shipley.test, 46
siblings *(*ancestry*)*, 6
summary.GGM, 48
summary.RICF, 48

weightGraph, *3*, *20*, *26*, 49