Package: snha (via r-universe)

September 13, 2024

Type Package

Title Creating Correlation Networks using St. Nicolas House Analysis

Version 0.2.1

Date 2024-07-06

Author Detlef Groth

Maintainer Detlef Groth <dgroth@uni-potsdam.de>

Description Create correlation networks using St. Nicolas House Analysis ('SNHA'). The package can be used for visualizing multivariate data similar to Principal Component Analysis or Multidimensional Scaling using a ranking approach. In contrast to 'MDS' and 'PCA', 'SNHA' uses a network approach to explore interacting variables. For details see 'Hermanussen et. al. 2021', <doi:10.3390/ijerph18041741>.

URL https://github.com/mittelmark/snha

BugReports https://github.com/mittelmark/snha/issues

Depends R (>= 3.5.0)

Imports MASS

Suggests knitr, rmarkdown

VignetteBuilder knitr

License MIT + file LICENSE

LazyData yes

Language en-US

Encoding UTF-8

NeedsCompilation no

Collate ll.R asgp.R priv.R snha.R mgraph.R

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

RemoteUrl https://github.com/mittelmark/snha

RemoteRef HEAD

RemoteSha 9bff0afa0c5f5d93ccb636fce6d8ab83ecdff29a

Contents

`snha-package` *snha package - association chain graphs from correlation networks*

Description

The snha package can be used to construct association chain graphs based on the St. Nicolas House Analysis (SNHA) algorithm as described in Groth et. al. 2019. and Hermanussen et. al. 2021.

Details

The package provides the following functions: Function for graph generation from data:

snha(data) applys the SNHA method on the data and returns a new snha graph object

S3 methods for snha graphs:

plot.snha(x) plots a snha graph

as.list.snha(x) return a list representation of a snha graph object

Utility functions:

snha_get_chains(g) returns the chains found by the algorithm as matrix

snha_graph2data(A) create for the given adjacency matrix some data with the appropiate correlations

snha_layout(g) calculate layout coordinates for the given graph or adjacency matrix

snha_ll(g,chain) calculate log-likelihood for the given chain of the snha graph

snha_rsquare(data,g) for given data and graph or adjacency matrix calculate linear model r-square value

Value

No return value

Author(s)

Detlef Groth <dgroth@uni-potsdam.de>

References

- Groth, D., Scheffler, C., & Hermanussen, M. (2019). Body height in stunted Indonesian children depends directly on parental education and not via a nutrition mediated pathway - Evidence from tracing association chains by St. Nicolas House Analysis. *Anthropologischer Anzeiger*, 76 No. 5 (2019), p. 445 - 451. doi:10.1127/anthranz/2019/1027
- Hermanussen, M., Assmann, & Groth, D. (2021). Chain Reversion for Detecting Associations in Interacting Variables - St. Nicolas House Analysis. *International Journal of Environmental Research and Public Health*. 18, 4 (2021). doi:10.3390/ijerph18041741.
- Novine, M., Mattsson, C. C., & Groth, D. (2021). Network reconstruction based on synthetic data generated by a Monte Carlo approach. *Human Biology and Public Health*, 3:26. doi:10.52905/hbph2021.3.26

Examples

```
library(MASS)
data(birthwt)
as=snha(birthwt[,-1])
plot(as)
as$theta
ls(as)
data(decathlon88)
head(decathlon88)
dec=snha(decathlon88,method="spearman",alpha=0.1)
plot(dec,layout='sam')
```

as.list.snha *return a list representation for an snha graph object*

Description

The function ‘as.list.snha‘ provides a S3 method to convert a snha graph object into a list object which can be for instance used to write a report into an XLSX file using the library openxlsx.

Usage

```
## S3 method for class 'snha'
as.list(x,...)
```

Arguments

x	snha graph object created with the snha function
...	additional arguments, delegated to the list command

Value

list object with the components: ’chains’ (the association chain), ’data’ (original data), ’theta’ (adjacency matrix, ’sigma’ (correlations), ’p.value’ (correlation p-values)

See Also

plot.snha, snha

Examples

```
data(swiss)
as=snha(swiss,method="spearman",alpha=0.1)
result=as.list(as)
ls(result)
result$settings
# can be writte as xlsx file for instance like:
# library(openxlsx)
# write.xlsx(result,file="some-result.xlsx")
```

decathlon88 *Men Decathlon data from the 1988 Olympics*

Description

A subset of data from the Decathlon from the 1988 Olympic games. Included are all athletes which finished with more than 7000 points.

Usage

```
decathlon88
```

Format

A data frame with 33 rows and 10 columns:

disc discus results in m

high high jump results in m

jave javelin through results in m

long long jump results in m

pole pole vault results in m

shot shot put results in m

X100 running speed over 100m in km/h

X110 running speed over 110m hurdles in km/h

X1500 running speed over 1500m in km/h

X400 running speed over 400m in km/h

Source

<https://en.wikipedia.org/wiki/Athletics_at_the_1988_Summer_Olympics_-_Men's_decathlon>

Examples

```
data(decathlon88)
head(decathlon88)
A=snha(decathlon88,method="spearman",alpha=0.1)
cols=rep("salmon",10)
cols[names(A$data) %in% c("jave","shot","disc","pole")]="skyblue"
plot(A,layout="sam",vertex.color=cols,vertex.size=8,cex=1.2,edge.width=5)
snha_rsquare(A)
```

`mgraph` *create a mgraph object, an adjacency matrix of a specific type*

Description

This is a function to create a few different graph types such as barabasi, random, cluster etc to test the snha algorithms.

Usage

```
mgraph(
  x=NULL,
  type="random",
  mode="directed",
  nodes=10,
  edges=12,
  m=1,
  k=3,
  p=NULL,
  power=1)
```

Arguments

`x`	either a adjacency matrix or an adjacency list, if given type is not used
`type`	graph type, one of 'angie', 'band', 'barabasi', 'circle', 'cluster', 'hubs', 'nicolas','random', 'regular' or 'werner', default: 'random'
`mode`	either 'undirected' or 'directed', default: 'directed'
`nodes`	number of nodes for a given type, default: 10
`edges`	number of edges for a given type, not used for type "barabasi", default: 12
`m`	number of edges added in each iteration of type is 'barabasi', can be values such as 1,2 or fractionla numbers such as 1.3 which means 70 percent single chain addition and 30 percent double chain additions, default: 1
`k`	the degree for a regular graph, or the number of clusters for a cluster graph or the number of hubs for a hubs graph, default: 2
`p`	the probabilty for edges if type is 'gnp',default=NULL
`power`	the power for preferential attachment if type is 'barabasi', 1 is linear preferential attachment, below one is sublinear, smaller hubs, 0 is no hubs, above 1 is super linear with super hubs, default: 1

Value

mgraph object, which contains an adjacency matrix and optional an attribute for a layout

See Also

snha, plot.snha

Examples

```
M <- matrix(0,nrow=7,ncol=7)
rownames(M)=colnames(M)=LETTERS[1:7]
M[c('A','B'),'C']=1
M['C','D']=1
M['D',c('E','F')]=1
M['E','F']=1
G=mgraph(M)
set.seed(125)
R = mgraph(type="random",nodes=8,edges=9)
summary(R)
A = mgraph(type="angie",nodes=8,edges=9)
A
C = mgraph(type="circle",nodes=8)
B = mgraph(type="band",nodes=8)
W=mgraph(type='werner')
W
B=mgraph(type='barabasi')
B
D=mgraph(type='barabasi',m=1.3)
D
N=mgraph(type='nicolas')
## Nicolas graph brings its own layout
plot(N,layout=attr(N,'layout'))
```

---

mgraph_accuracy *quality measures for a predicted graph*

Description

The function ‘mgraph_accuracy‘ measures the prediction quality of a predicted graph in comparison to a known true one. The graph comparisons are done on the basis of undirected graphs only. Directed graphs are converted to undirected internally.

The most used accuracy measure to balance the different measures is the F-score where:

- the F0.5 measure (beta=0.5) gives more weight on precision, less weight on sensitivity/recall.
- the F1 measure (beta=1.0) balances the weight on precision and sensitivity/recall.
- the F2 measure (beta=2.0) gives less weight on precision, more weight on sensitivity/recall

Usage

```
mgraph_accuracy(g.true,g.pred)
```

Arguments

g.true	snha, mgraph or or adjacency matrix of a the true graph
g.pred	snha, mgraph or or adjacency matrix of a the predicted graph

Value

returns list object with various accuracy measures, such as Sens(itivity), Spec(ificity), Prec(ision), Acc(uracy), Non-information-rate (NIR), balanced classification rate (BCR), F1 measure and Mathhews correlation coefficient (MCC)

Examples

```
ang=mgraph(type="angie",nodes=12,edges=18)
data=snha_graph2data(ang)
pred=snha(t(data))$theta
round(unlist(mgraph_accuracy(ang,pred)),2)
#  using vectors
tvec=c(1,1,1,0,0,0)
pvec=c(1,1,0,1,0,0)
table(tvec,pvec)
mgraph_accuracy(tvec,pvec)
```

---

mgraph_autonames *create names for nodes and other data structures*

Description

This function aids in creating standard node labels for graphs.

Usage

```
mgraph_autonames(
  n,
  prefix=NULL)
```

Arguments

n	how many labels
prefix	the node prefix, per defaultif not given LETTERS with numbers will be used, default: NULL

#

Examples

```
mgraph_autonames(12)
mgraph_autonames(12,LETTERS[1:4])
mgraph_autonames(12,"R")
```

mgraph_d2u *create an undirected graph out of a directed graph*

Description

This function gets an directed graph and convertes all edges from directed ones to undirected ones. The number of edges should stay the same, the edge sign (+ or -) stays the same.

Usage

```
mgraph_d2u(g)
```

Arguments

g	a mgraph object or an adjacency matrix

#

Examples

```
A=mgraph(type="angie",nodes=4,edges=4)
A
U=mgraph_d2u(A)
U
```

mgraph_degree *return the number of edges for each node*

Description

This function returns the number of edges adjacent to every node, for directed graphs it will as well return the in and out going edges.

Usage

```
mgraph_degree(
  g,
  mode='undirected')
```

Arguments

g	a mgraph object
mode	how should the graph been analyzed, either 'undirected', 'out' or 'in', default: 'undirected'

#

Examples

```
A=mgraph(type="regular",nodes=8,k=3)
mgraph_degree(A)
mgraph_degree(A,mode="out")
mgraph_degree(A,mode="in")
```

`mgraph_lmc` *using linear models to check the create snha graph*

Description

This function checks a given snha graph against its own data using linear models. Edges are removed if they do not add more than 2 percent of the adjusted R-square value. The function might be only called directly if the user would like to change the default values for del and add. The function snha uses defaults of 0.02 for del(eting) and 0.05 for add(ing) values.

Usage

```
mgraph_lmc(x,del=0.02,add=NULL,method='pearson')
```

Arguments

`x`	snha graph object
`del`	r-square threshold to delete edges, edges not adding more than this value to the linear model to the target node will be deleted, default: 0.02
`add`	r-square threshold to add edges, if NULL no edges will be added, recommended value is 0.05, default: NULL
`method`	The method which was used to generate a graph, if spearman or mutual information were given data are rank transformed, default: 'pearson'

Value

returns adjacency matrix where edges migh be removed if they are not adding explanation to the model

Examples

```
set.seed(123)
B=mgraph(type="barabasi",m=2,nodes=6)
data=t(snha_graph2data(B))
aa=snha(data)
aa$theta
unlist(mgraph_accuracy(B,aa$theta))
ab=mgraph_lmc(aa)
unlist(mgraph_accuracy(B,ab))
ab
aa$theta-ab
ac=mgraph_lmc(aa,add=0.02)
unlist(mgraph_accuracy(B,ac))
```

mgraph_lms *linear model backward variable selection*

Description

This function is a simple version of linear model building using backward variable selection. At each step the variable with the lowest decrease value in the adjusted R-square value is removed from the model and the r-square value is stored in the return matrix.

Usage

```
mgraph_lms(x)
```

Arguments

x	a data matrix or data frame

Value

returns matrix with the adjusted r-square values attributed by the variables

Examples

```
data(swiss)
round(mgraph_lms(swiss)*100,2)
W = mgraph(type='werner')
data=t(snha_graph2data(W))
round(mgraph_lms(data)*100,2)
```

mgraph_nd *network deconvolved data matrix using the algorithm of Feizi et. al. 2013*

Description

This function is a R implementation for "A General Method to Distinguish Direct Dependencies over Networks" by Feizi et. al (2013). For the Matlab code look here: http://compbio.mit.edu/nd/code/ND.m

Usage

```
mgraph_nd(x,beta=0.99,alpha=1,control=FALSE)
```

Arguments

x	symmetric relevance matrix, where high similarity between nodes is expressed with high values such as in an correlation matrix
beta	scaling paramater, the largest absolute eigenvalue is mapped to beta, values should be between 0 and 1, default: 0.99
alpha	fraction of edges of the observed dependency matrix to be kept, should be between 0 (none) and 1 (all), default: 1
control	if FALSE only direct weights are displayed, if TRUE also non-observed interactions are displayed, default: FALSE

Value

returns matrix with deconvoluted relevance matrix

Author(s)

- @2013 KELLIS-LAB, Soheil Feizi, Matlab code
- @2021 Detlef Groth, University of Potsdam, R code

References

- Feizi, S., Marbach, D., Medard, M., & Kellis, M. (2013). Network deconvolution as a general method to distinguish direct dependencies in networks. Nature biotechnology, 31(8), 726-733. DOI 10.1038/nbt.2635

Examples

```
W=mgraph(type="werner")
W
data=snha_graph2data(W)
C=cor(t(data))
round(C,2)
round(mgraph_nd(C,beta=0.9,alpha=0.3),2)
# manually prepare the deconvoluted matrix
D=mgraph_nd(C,beta=0.9,alpha=0.3)
diag(D)=1
par(mfrow=c(2,2))
plot(W)
plot(snha(t(data)))
plot(snha(t(data),nd=TRUE))
plot(snha(D))
```

mgraph_nodeColors *create node colors for directed graphs*

Description

This function simplifies automatic color coding of nodes for directed graphs. Nodes will be colored based on their degree properties, based on their incoming and outcoming edges.

Usage

```
mgraph_nodeColors(g,
    col=c("skyblue","grey80","salmon")
)
```

Arguments

g	a mgraph object or an adjacency matrix or an adjacency list
col	default colors for nodes with only incoming, in- and outgoing and only outgoing edges, default: c("skyblue","grey80","salmon")

#

Examples

```
par(mfrow=c(1,2),mai=rep(0.1,4))
A=mgraph(type="random",nodes=6,edges=8)
cols=mgraph_nodeColors(A)
mgraph_degree(A,mode="in")
mgraph_degree(A,mode="out")
cols
plot(A, layout="star")
plot(A, layout="star",vertex.color=cols)
```

mgraph_trf *checking possible chains with 3 nodes for triad structures*

Description

Experimental:

This function checks a given snha graph for possible triad structures which were not recognized by the original algorithm. The method checks all nodes within path length 2 for a possible edge by comparing the ratio of the correlation coefficients. The function snha uses defaults of 0.02 for del(eting) and 0.05 for add(ing) values.

Usage

```
mgraph_trf(x,frac=1.2)
```

Arguments

x	snha graph object
frac	amount which the absolute correlation exceeds in compariuson of the product of the other two correlations in the possible triad, value should be larger than 1, default: 1.2

Value

returns snha graph object with added triads to chains and to the adjacency matrix

Examples

```
set.seed(124)
W=mgraph(type="werner")
data=t(snha_graph2data(W))
aa=snha(data)
aa$theta
unlist(mgraph_accuracy(W,aa$theta))
# remove false positive by linear model
ab=mgraph_lmc(aa)
aa$theta=ab
aa=mgraph_trf(aa)
unlist(mgraph_accuracy(W,aa$theta))
```

---

mgraph_u2d *reate a directed graph out of an undirected graph*

Description

This function creates a directed graph from an undirected one by the given input nodes. Input nodes can be chosen by names or a number for random selection of input nodes will be given. Input nodes will have at least shortest path distance to other input nodes of pathlength two. Selected input nodes will draw in each iteration outgoing edges to other nodes in the nth iteration neighborhood. The input nodes will alternatively select the next edges on the path to not visited nodes. All edges will be only visited onces.

Usage

```
mgraph_u2d(
    g,input=2,negative=0.0,shuffle=FALSE
)
```

Arguments

g	mgraph object created with mgraph
input	number or names of input nodes in the graph, if number of input nodes is smaller than number of components, for each component one input node is automatically created

negative	proportion of inhibitive associations in the network value between 0 and 1 are acceptable, Default 0.0
shuffle	should just the edge directions beeing shuffled, if TRUE the graph will be very random without a real structure or chains of associations, default: FALSE

#

Examples

```
par(mfrow=c(2,2),mai=rep(0.1,4))
G=mgraph(type="angie",nodes=7,edges=9)
mgraph_degree(G,mode='in')
U=mgraph_d2u(G)
H=mgraph_u2d(U,input=c("G","E"))
identical(G,H)
I=mgraph_u2d(U,shuffle=TRUE)
identical(G,I)
lay=snha_layout(G)
plot(G,layout=lay,vertex.color=mgraph_nodeColors(G))
plot(U,layout=lay)
plot(H,layout=lay,vertex.color=mgraph_nodeColors(H))
plot(I,layout=lay,vertex.color=mgraph_nodeColors(I))
```

---

plot.snha *display network or correlation matrices of snha graphs*

Description

The function 'plot.snha' provides a simple display of network graphs or correlation matrices using filled circles (vertices) to represent variables and edges which connect the vertices with high absolute. correlation values. Positive correlations are shown in black, negative correlations are shown in red. For more information see the details section.

Usage

```
## S3 method for class 'snha'
plot(
 x,
 type = "network",
 layout = "circle",
 vertex.color = "salmon",
 cex = 1,
 vertex.size = 5,
 edge.width = 2,
 edge.color = c("grey70", "red"),
 edge.text = NULL,
 edge.cex = 0.8,
 edge.pch = 0,
 noise = FALSE,
```

```
 hilight.chain = NULL,
 chain.color = c("black", "red"),
 star.center = NULL,
 plot.labels = TRUE,
 lty = 1,
 threshold = c(0.25, 0.5, 0.75),
 interactive = FALSE,
 ...
)
```

Arguments

x	snha graph object usually created with the 'snha' function or an adjacency matrix
type	character string specifying the plot type either 'network' or ' cor', default: 'network'
layout	graph layout for plotting one of 'circle', 'sam', 'samd', 'grid', 'mds', 'mdsd', 'star', default: 'circle'
vertex.color	default color for the vertices, either a single value, all vertices have hen this color or a vector of values, for different colors for the nodes, default: 'salmon'
cex	size of the vertex labels which are plotted on the vertices, default: 1
vertex.size	number how large the vertices should be plotted, default: 5
edge.width	number on how strong the edges should be plotted, if edge.width=0, then the number is based on the correlation values, default: 2
edge.color	color to be plotted for edges. Usually vector of length two. First color for positive correlations, second color for negative correlations. Default: c('grey','red')
edge.text	optional matrix to give edge labels, default: NULL
edge.cex	character expansion for edge labels, default: 0.8
edge.pch	plotting character which should be placed below the edge.text, default: 0
noise	should be noise added to the layout. Sometimes useful if nodes are too close. Default: FALSE
hilight.chain	which chain should be highlighted, default: NULL (no chain highlight)
chain.color	which color for chain edges, default: black
star.center	the centered node if layout is 'start', must be a character string for the node name, default: NULL
plot.labels	should node labels plotted, default: TRUE
lty	line type for standard edges in the graph, default: 1
threshold	cutoff values for bootstrap probabilities for drawing edges as dotted, broken lines and solid lines, default: c(0.25,0.5,0.75)
interactive	switch into interactive mode where you can click in the graph and move nodes with two clicks, first selecting the node, second click gives the new coordinates for the node, default: FALSE
...	currently not used

Details

This is a plot function to display networks or correlation matrices of 'snha' graph objects. In case of bootstrapping the graph by using the 'snha' function with the 'prob=TRUE' option lines in style full, broken and dotted lines are drawn if they are found in more than 75, 50 or 25 percent of all re-samplings. You can change these limits by using the 'threshold' argument.

Value

returns the layout of the plotted network or NULL if type is 'corrplot' (invisible)

Examples

```
data(swiss)
sw.g=snha(swiss,method='spearman')
sw.g$theta
round(sw.g$sigma,2)
plot(sw.g,type='network',layout='circle')
plot(sw.g,type='network',layout='sam')
plot(sw.g,type='corplot')
# adding correlation values
plot(sw.g,edge.text=round(sw.g$sigma,2),edge.cex=1.2,edge.pch=15)
sw.g=snha(swiss,method='spearman',prob=TRUE)
sw.g$theta
sw.g$probabilities
plot(sw.g,type='network',layout='sam')
sw.g$chains
# plot chains for a node
plot(sw.g,layout="sam",lty=2,hilight.chain="Infant.Mortality",
 edge.width=3,edge.color=c("black","red"))
# an example for an adjacency matrix
M=matrix(rbinom(100,1, 0.2),nrow=10,ncol=10)
diag(M)=0
colnames(M)=rownames(M)=LETTERS[1:10]
plot.snha(M)
```

---

snha *Initialize a snha object with data.*

Description

The main entry function to initialize a snha object with data where variables are in columns and items are in rows. If you like to use the network deconvolution method of Feizi et al. instead of the data submit the deconvoluted matrix. As methods either correlation methods such as pearson, spearman or kendall or mutual information (mi) can be used. The latter method as well covers non-linear linear relationships.

Usage

```
snha(
  data,
  alpha=0.05,
  method='pearson',
  threshold=0.01,
  check.singles=FALSE,
  prob=FALSE,
  prob.threshold=0.2,
  prob.n=25,
  nd=FALSE,
  lmc=FALSE,
  lma=FALSE)
```

Arguments

data	a data frame where network nodes are the row names and data variables are in the columns.
alpha	confidence threshold for p-value edge cutting after all chains were generated, default: 0.05.
method	method to calculate correlation/association values, can be 'pearson', 'spearman' or 'kendall', 'mi', default: 'pearson'.
threshold	R-squared correlation coefficient threshold for which r-square values should be used for chain generation, r=0.1 is r-square of 0.01, default: 0.01.
check.singles	should isolated nodes connected with sufficient high R^2 and significance, default: FALSE.
prob	should be probabilities computed for each edge using bootstrapping. Only in this case the parameters starting with prob are used, default: FALSE
prob.threshold	threshold to set an edge, a value of 0.5 means, that the edge must be found in 50% of all samplings, default: 0.2
prob.n	number of bootstrap samples to be taken, default: 25
nd	perfom network deconvolution due to the method of Feizi et. al. 2013, default: FALSE
lmc	perfom a linear model check, edges which does not improve the node model by adjacent edges more than 0.02 are removed, default: FALSE
lma	perfom a linear model check, edges which does improve the node model by adjacent edges with more than 0.02 R-square are added, works only if lmc is TRUE,default: FALSE

Value

A snha graph data object with the folling components:

chains association chains building the graph

data representing the original input data

p.values matrix with p-values for the pairwise correlations

probabilities in case of re-samplings, the proportion how often the chain was found

sigma correlation matrix used for the algorithm

theta adjacency matrix found by the SNHA method

See Also

plot.snha

Examples

```
data(swiss)
sw.g=snha(swiss,method='spearman')
# what objects are there?
ls(sw.g)
sw.g$theta
round(sw.g$sigma,2)
sw.g=snha(swiss,method='spearman',check.singles=TRUE,prob=TRUE)
sw.g$theta
sw.g$probabilities
sw.m=snha(swiss,method="mi")
sw.m$theta
par(mfrow=c(1,2))
plot(sw.g)
plot(sw.m)
```

---

snha_corrplot *visualize a matrix of pairwise correlations*

Description

This function returns xy coordinates for a given input adjacency matrix or snha graph. It is useful if you like to plot the same set of nodes with different edge connections in the same layout.

Usage

```
snha_corrplot(
   x,
   text.lower=FALSE,
   text.upper=FALSE,
   pch.minus=19,
   pch.plus=19,
   xtext=NULL,
   cex=1.0,
   ...)
```

Arguments

x	matrix with pairwise correlations
text.lower	should in the lower diagonal the correlation coefficient be shown, default: TRUE
text.upper	should in the upper diagonal the correlation coefficient be shown, default: FALSE
pch.minus	the plotting symbol for negative correlations, default: 19
pch.plus	the plotting symbol for positive correlations, default: 19
xtext	labels which should be placed at the bottom of the plot, default: NULL
cex	character expansion for text and correlation symbols, default: 1
...	arguments delegated to the plot function

Examples

```
data(swiss)
sw=swiss
colnames(sw)=abbreviate(colnames(swiss),6)
options(warn=-1) # avoid spearman warnings
snha_corrplot(cor(sw,method="spearman"),cex.sym=8,text.lower=TRUE)
options(warn=0)
```

---

snha_get_chains *Return the chains of a snha graph as data frame*

Description

This is a utility function to return the chains which constructs the graph as a matrix.

Usage

```
snha_get_chains(graph)
```

Arguments

graph	a snha graph object

Value

matrix with one chain per row, shorter chains are filled up with empty strings

Examples

```
data(swiss)
sw.g=snha(swiss)
snha_get_chains(sw.g)
```

`snha_graph2data` *create correlated data for the given adjacency matrix representing a directed graph or an undirected graph*

Description

This function is a short implementation of the Monte Carlo algorithm described in Novine et. al. 2022.

Usage

```
snha_graph2data(
  A,
  n=100,
  iter=50,
  val=100,
  sd=2,
  prop=0.025,
  noise=1,
  method="mc"
  )
```

Arguments

`A`	an adjacency matrix
`n`	number of values, measurements per node, default: 100
`iter`	number of iterations, default: 50
`sd`	initial standard deviation, default: 2
`val`	initial node value, default: 100
`prop`	proportion of the target node value take from the source node, default: 0.025
`noise`	sd for the noise value added after each iteration using rnorm function with mean 0, default: 1
`method`	method for data generation, either 'mc' for using Monte Carlo simulation or 'pc' for using a precision matrix, default: 'mc'

Value

matrix with the node names as rows and samplings in the columns

References

- Novine, M., Mattsson, C. C., & Groth, D. (2021). Network reconstruction based on synthetic data generated by a Monte Carlo approach. *Human Biology and Public Health*, 3:26. doi:10.52905/hbph2021.3.26

Examples

```
opar=par(mfrow=c(1,2),mai=rep(0.2,4))
A=matrix(0,nrow=6,ncol=6)
rownames(A)=colnames(A)=LETTERS[1:6]
A[1:2,3]=1
A[3,4]=1
A[4,5:6]=1
A[5,6]=1
plot.snha(A,layout="circle");
data=snha_graph2data(A)
round(cor(t(data)),2)
P=snha(t(data))
plot(P,layout="circle")
par(opar)
```

`snha_layout` *Determine graph layouts*

Description

This function returns xy coordinates for a given input adjacency matrix or snha graph. It is useful if you like to plot the same set of nodes with different edge connections in the same layout.

Usage

```
snha_layout(
   A,
   mode='sam',
   method='pearson',
   noise=FALSE,
   star.center=NULL,
   interactive=FALSE)
```

Arguments

`A`	an adjacency matrix or an snha graph object
`mode`	character string for the layout type, can be either 'mds' (mds on graph using shortest paths), 'mdsd' (mds on data) 'sam' (sammon on graph), 'samd' (sammon on data), 'circle', 'grid' or 'star', default: 'sam'
`method`	method for calculating correlation distance if mode is either 'mdsd' or 'samd', default: 'pearson'
`noise`	should some noise be added, default: FALSE
`star.center`	the centered node if layout is 'star', must be a character string for the node name, default: NULL
`interactive`	switch into interactive mode where you can click in the graph and move nodes with two clicks, first selecting the node, second click gives the new coordinates for the node, default: FALSE

Value

matrix with x and y columns for the layout

Examples

```
data(swiss)
sw.s=snha(swiss,method='spearman')
sw.p=snha(swiss,method='pearson')
lay=snha_layout(sw.s,mode='sam')
plot(sw.s,layout=lay)
plot(sw.p,layout=lay)
plot(sw.s,layout='star',star.center='Education')
rn1=rnorm(nrow(swiss))
nswiss=cbind(swiss,Rn1=rn1)
plot(snha(nswiss,method='spearman'),layout='sam')
plot(snha(nswiss,method='spearman'),layout='samd',
  vertex.size=2,vertex.color='beige')
```

---

snha_ll *log-likelihood for the given snha graph and the given chain*

Description

This function returns the log-likelihood for the given snha graph and the given chain. If the 'block.p.value' is lower than 0.05 than that the chain is not sufficient to capture the variable dependencies, p-values above 0.05 indicate a good coverage of the chain for the linear dependencies between the nodes.

Usage

```
snha_ll(graph,chain=NULL)
```

Arguments

graph	a snha graph object
chain	a chain object of a snha graph, if not given a data frame with the values is returned for all chains, default: NULL

Value

list with the following components: 'll.total', 'll.chain', 'll.rest', 'll.block', data frame 'df' with the columns 'chisq', 'p.value', 'block.df', 'block.ch', 'block.p.value'. If chain is not given an overall summary is made for all chains an returned as data frame.

Examples

```
data(swiss)
sw.g=snha(swiss)
snha_ll(sw.g,sw.g$chain$Catholic)
head(snha_ll(sw.g))
```

---

snha_mi *mutual information for two vectors or a matrix*

Description

This function computes the mutual information between two numerical variables or for all pairs of variables if a matrix is given. The normalization is actual value divided by the the geometric mean: I'(X,Y) = I(X,Y)/sqrt(I(X)*I(Y))

Usage

```
snha_mi(x,y=NULL,breaks=4,coc=FALSE,gnorm=FALSE,norm=FALSE)
```

Arguments

x	either a binned table, a matrix or data.frame or a numerical vector
y	numerical vector, required if x is a vector, otherwise ignored,default:NULL
breaks	number of breaks to create a binned table, default: 4
coc	Coefficient of constrains, Cxy=Ixy/Hy and Cyx=Ixy/Hx, this measure is asymetric,default:FALSE
gnorm	mutual information should be normalized by using the geometric mean Ixy=Ixy/sqrt(Ix*Iy),default:FALSE
norm	divide every MI value by the MI value for the variable, this measure is as well asymetric,default:FALSE

Value

mutual information value as scalar if input is table or two vectors or as matrix if input is matrix or data.frame

Examples

```
rn1=rnorm(100,mean=10,sd=1);
rn2=rn1+0.5*rnorm(100)
cor(rn1,rn2) # high
cor(rn1,sample(rn2)) #random
snha_mi(rn1,rn2) # high
snha_mi(rn1,sample(rn2)) #random
snha_mi(rn1,rn2,breaks=4)
snha_mi(rn1,rn2,breaks=7)
data(swiss)
round(snha_mi(swiss),2)
```

```
round(snha_mi(swiss,gnorm=TRUE),2)
as=snha(snha_mi(swiss,gnorm=TRUE))
plot(as)
```

---

snha_rsquare *linear model based r-square values for given data and graph*

Description

The function 'snha_rsquare' calculates for given data and a graph the covered r-squared values by a linear model for each node. The linear model predicts each node by an additive mode using it's neighbor nodes in the graph.

Usage

```
snha_rsquare(data,graph=NULL)
```

Arguments

data	data matrix or data frame where variables are in columns and samples in rows or a snha graph
graph	graph object or adjacency matrix of an (un)directed graph, not needed if data is a snha graph, default: NULL.

Value

vector of rsquare values for each node of the graph

Examples

```
# random adjacency matrix
A=matrix(rbinom(100,1, 0.2),nrow=10,ncol=10)
diag(A)=0
colnames(A)=rownames(A)=LETTERS[1:10]
# random data
data=matrix(rnorm(1000),ncol=10)
colnames(data)=colnames(A)
snha_rsquare(data,A)
# real data
data(swiss)
sw.s=snha(swiss,method='spearman')
rsqs=snha_rsquare(sw.s)
plot(sw.s,main=paste("r =",round(mean(rsqs,2))),
   layout='star',star.center='Examination')
# some colors for r-square values
vcols=paste("grey",seq(80,40,by=-10),sep="")
scols=as.character(cut(snha_rsquare(swiss,sw.s$theta),
   breaks=c(0,0.1,0.3,0.5,0.7,1),labels=vcols))
plot(sw.s,main=paste("r =",round(mean(snha_rsquare(swiss,sw.s$theta)),2)),
```

```
vertex.color=scols ,layout='star',star.center='Examination',
vertex.size=10,edge.color=c('black','red'),edge.width=3)
```

Index