Package 'qgraph'

May 8, 2013

Type Package

Title Network representations of relationships in data

Version 1.2.2

Date 2013-05-08

Author Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp,Verena D. Schmittmann and Denny Borsboom

Maintainer Sacha Epskamp <qgraph@sachaepskamp.com>

Depends R (>= 2.15.0)

Suggests RSVGTipsDevice, fdrtool, lavaan, sem, rpanel

Imports psych, ellipse, lavaan, sem, plyr, methods, Hmisc, igraph,jpeg, png, colorspace

ByteCompile yes

Description The qgraph package can be used to visualize data as networks.

URL http://sachaepskamp.com/qgraph

License GPL-2

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-05-08 18:38:17

R topics documented:

`qgraph-package` *qgraph*

Description

This package can be used to visualize psychometric data. See `qgraph` for the main function.

Details

Package:	qgraph
Type:	Package
Version:	1.0
Date:	2010-08-16
License:	GPL-2
LazyLoad:	yes

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

Maintainer: Sacha Epskamp<s.epskamp@uva.nl>

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

`as.igraph.qgraph` *Converts qgraph object to igraph object.*

Description

This function converts the output of `qgraph` to an 'igraph' object that can be used in the igraph package (Csardi & Nepusz, 2006)

Usage

```
## S3 method for class 'qgraph'
as.igraph(object, attributes = TRUE)
```

Arguments

`object`	A "qgraph" object
`attributes`	Logical, should graphical attributes also be transferred?

Author(s)

Sacha Epskamp <qgraph@sachaepskamp.com>

References

Csardi G, Nepusz T (2006). The igraph software package for complex network research, InterJournal, Complex Systems 1695. http://igraph.sf.net

big5 *Big 5 dataset*

Description

This is a dataset of the Big 5 personality traits that will be used in talks and the paper. It is a measurement of the Dutch translation of the NEO-PI-R on 500 first year psychology students (Dolan, Oort, Stoel, Wicherts, 2009).

Usage

```
data(big5)
```

Format

The format is: num [1:500, 1:240] 2 3 4 4 5 2 2 1 4 2 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr [1:240] "N1" "E2" "O3" "A4" ...

References

Hoekstra, H. A., Ormel, J., & De Fruyt, F. (2003). NEO-PI-R/NEO-FFI: Big 5 persoonlijkheidsvragenlijst. Handleiding \[Manual of the Dutch version of the NEO-PI-R/NEO-FFI\]. Lisse, The Netherlands: Swets and Zeitlinger.

Dolan, C. V., Oort, F. J., Stoel, R. D., & Wicherts, J. M. (2009). Testing measurement invariance in the target rotates multigroup exploratory factor model. Structural Equation Modeling, 16, 295<96>314.

big5groups *Big 5 groups list*

Description

This is the groups list of the big 5 data. It is a measurement of the Dutch translation of the NEO-PI-R on 500 first year psychology students (Dolan, Oort, Stoel, Wicherts, 2009).

Usage

```
data(big5groups)
```

Format

The format is: List of 5 $ Neuroticism : num [1:48] 1 6 11 16 21 26 31 36 41 46 ... $ Extraversion : num [1:48] 2 7 12 17 22 27 32 37 42 47 ... $ Openness : num [1:48] 3 8 13 18 23 28 33 38 43 48 ... $ Agreeableness : num [1:48] 4 9 14 19 24 29 34 39 44 49 ... $ Conscientiousness: num [1:48] 5 10 15 20 25 30 35 40 45 50 ...

References

Hoekstra, H. A., Ormel, J., & De Fruyt, F. (2003). NEO-PI-R/NEO-FFI: Big 5 persoonlijkheidsvragenlijst. Handleiding [Manual of the Dutch version of the NEO-PI-R/NEO-FFI]. Lisse, The Netherlands: Swets and Zeitlinger.

Dolan, C. V., Oort, F. J., Stoel, R. D., & Wicherts, J. M. (2009). Testing measurement invariance in the target rotates multigroup exploratory factor model. Structural Equation Modeling, 16, 295<96>314.

`centrality` *Centrality statistics of graphs*

Description

This function can be used on the output of `qgraph` to compute the node centrality statistics for weighted graphs proposed by Opsahl, Agneessens and Skvoretz (2010).

Usage

```
centrality(graph, alpha = 1, posfun = abs)
```

Arguments

`graph`	A "qgraph" object obtained from `qgraph`
`alpha`	The tuning parameter. Defaults to 1.
`posfun`	A function that converts positive and negative values to only positive. Defaults to the absolute value.

Details

This function computes and returns the in and out degrees, closeness and betweenness as well as the shortest path lengths and shortest paths between all pairs of nodes in the graph. For more information on these statistics, see Opsahl, Agneessens and Skvoretz (2010).

These statistics are only defined for positive edge weights, and thus negative edge weights need to be transformed into positive edge weights. By default, this is done by taking the absolute value.

The algorithm used for computing the shortest paths is the well known "Dijkstra's algorithm" (Dijkstra, 1959). The algorithm has been implemented in R, which can make this function take several minutes to run for large graphs (over 100 nodes). A future version of qgraph will include a compiled version to greatly speed up this function.

Value

A list containing:

`OutDegree`	A vector containing the outward degree of each node.
`InDegree`	A vector containing the inward degree of each node.

`Closeness`	A vector containing the closeness of each node.
`Betweenness`	A vector containing the betweenness of each node
`ShortestPathLengths`	A matrix containing the shortest path lengths of each pairs of nodes. These path lenghts are based on the inverse of the absolute edge weights raised to the power alpha.
`ShortestPaths`	A matrix of lists containing all shortest path lengths between all pairs of nodes. Use double square brackets to index. E.g., if the list is called 'res', res$ShortestPaths[[i,j]] gives a list containing all shortest paths between node i and j.

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Opsahl, T., Agneessens, F., Skvoretz, J. (2010). Node centrality in weighted networks: generalizing degree and shortest paths. Soc Netw. 32:245–251.

Dijkstra, E.W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

See Also

qgraph

Examples

```
set.seed(1)
adj <- matrix(sample(0:1,10^2,TRUE,prob=c(0.8,0.2)),nrow=10,ncol=10)
Q <- qgraph(adj)

centrality(Q)
```

---

plot.qgraph *Plot method for "qgraph"*

Description

This function is exactly the same as qgraph

Usage

```
## S3 method for class 'qgraph'
plot(x, ...)
```

Arguments

x	A "qgraph" object
...	Arguments send to qgraph

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

---

print.qgraph *Print edgelist*

Description

This function prints the edgelist of a "qgraph" object

Usage

```
## S3 method for class 'qgraph'
print(x, ...)
```

Arguments

x	A "qgraph" object
...	These arguments are not used

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

See Also

qgraph

---

qgraph *qgraph*

Description

This is the main function of qgraph. It is typically used to visualize the relationship between several variables as a network. This function can be used to create various types of networks based on a number of input options. Please see the details section for more detail.

Usage

```
qgraph( input, ... )
```

Arguments

`input`	Can be either a weights matrix or an edgelist. Can also be an object of class "sem" (sem), "mod" (sem), "lavaan" (lavaan), "principal" (psych), "loadings" (stats), "factanal" (stats), "graphNEL" (Rgraphviz) or "pcAlgo" (pcalg)
`...`	Any additional arguments described below. Also a list with class "qgraph" can be added that contains any of these arguments (this is returned invisibly by the function)

Details

The qgraph function has a lot of arguments. Mostly the default values of these work well. Because of the amount of arguments the usage of the qgraph function has been reduced by using the ... method for clarity. This does mean that arguments need to be specified by using their exact name. For instance, to specify color="red" you can not use col="red".

Below is a complete list of all the arguments that can be used in qgraph and a detailed guide on how the function can be used.

Value

qgraph returns (invisibly) a 'qgraph' object that contains all the arguments used, with the exception of the 'layout' argument which is set to the final layout used in the graph. This can then be sent to a new qgraph function to use the same arguments in the new plot.

One of the values returned is 'layout.orig', the original (not rescaled) layout, which needs to be used when using constraint layouts.

Important additional arguments

gui Logical, calls `qgraph.gui` if TRUE to open GUI window.

layout This argument controls the layout of the graph. "circle" places all nodes in a single circle, "circular" gives a circular layout in which each group is put in separate circles and "spring" gives a force embedded layout. It also can be a matrix with a row for each node and x and y coordinates in the first and second column respectively. Defaults to "circular" in weighted graphs without a groups list, "groups" in weighted graphs with a groups list, and "spring" in unweighted graphs. Can also be a function from the igraph package.

groups An object that indicates which nodes belong together. Can be a list in which each element is a vector of integers identifying the numbers of the nodes that belong together, or a factor.

minimum Edges with absolute weights under this value are omitted. Defaults to 0 for graphs with less than 50 nodes or 0.1 for larger graphs.

maximum qgraph regards the highest of the maximum or highest absolute edge weight as the highest weight to scale the edge widths too. To compare several graphs, set this argument to a higher value than any edge weight in the graphs (typically 1 for correlations).

cut In weighted graphs, this argument can be used to cut the scaling of edges in width and color saturation. Edges with absolute weights over this value will have the strongest color intensity and become wider the stronger they are, and edges with absolute weights under this value will have the smallest width and become vaguer the weaker the weight. If this is set to NULL, no cutoff is used and all edges vary in width and color. Defaults to NULL for graphs with less then 50 nodes and 0.3 to larger graphs.

details Logical indicating if minimum, maximum and cutoff score should be printed under the graph. Defaults to FALSE.

Output arguments

mar A vector of the form c(bottom, left, top, right) which gives the margins. Works similar to the argument in par(). Defaults to c(3,3,3,3)

filetype A character containing the file type to save the output in. "R" and "x11" outputs in a new R window ("x11" is preferred for Rstudio), "pdf" creates a pdf file. "svg" creates a svg file (requires RSVGTipsDevice). "tex" creates LaTeX code for the graph (requires tikzDevice). 'jpg', 'tiff' and 'png' can also be used. If this is given any other string (e.g. filetype="") no device is opened. Defaults to 'R' if the current device is the NULL-device or no new device if there already is an open device.

filename Name of the file without extension

width Width of the plot, in inches

height Height of the plot, in inches

normalize Logical, should the plot be normalized to the plot size. If TRUE (default) border width, vertex size, edge width and arrow sizes are adjusted to look the same for all sizes of the plot, corresponding to what they would look in a 7 by 7 inches plot if normalize is FALSE.

standAlone Logical. If filetype="tex" this argument can be used to choose between making the output a standalone LaTeX file or only the codes to include the graph.

Graphical arguments

color A vector with a color for each element in the groups list, or a color for each node. Defaults to the background color ("bg" argument, which defaults to "white") without groups list and rainbow(length(groups)) with a groups list.

pastel Logical, should default colors (for groups or edge equality constraints) be chosen from pastel colors? If TRUE then `rainbow_hcl` is used.

rainbowStart A number between 0 and 1 indicating the offset used in rainbow functions for default node coloring.

legend Logical value indicating if a legend should be plotted. Defaults to TRUE if a groups object or nodeNames is supplied

legend.cex Scalar of the legend. defaults to 1

nodeNames Names for each node, can be used to plot a legend next to the plot that links the node labels to node names.

legend.mode Character, either `"groups"` indicating the legend should be based on the `groups` object, or `"names"` indicating the legend should be based on the `nodeNames` object.

GLratio Relative size of the graph compared to the layout. Defaults to 2.5

vsize A value indicating the size of the nodes (horizontal if shape is "rectangle". Can also be a vector of length 2 (nodes are scaled to degree) or a size for each node. Defaults to: max((-1/72)*(nNodes)+5.35,1)

vsize A value indicating the vertical size of the nodes where the shape is "rectangle". Can also be a vector of length 2 (nodes are scaled to degree) or a size for each node. Defaults to the value of 'vsize'. If 'vsize' is not assigned this value is used as a scalar to 'vsize' (e.g., `vsize2 = 1/2` would result in rectangled nodes where the height is half the default width)

esize Size of the largest edge (or what it would be if there was an edge with weight maximum). Defaults to: max((-1/72)*(nNodes)+5.35,1) for weighted graphs and 2 for unweighted graphs. In directed graphs these values are halved.

labels If FALSE, no labels are plotted. If TRUE, order in weights matrix is used as labels. This can also be a vector with a label for each node. Defaults for graphs with less than 20 nodes to a 3 character abbreviation of the columnames and rownames if these are identical or else to TRUE. If a label contains an asterisk (e.g. "x1*") then the asterisk will be omitted and the label will be printed in symbol font (use this for Greek letters). Can also be a list with a label as each element, which can be expressions for more advanced mathematical annotation.

label.cex Scalar on the label size.

label.prop Controls the proportion of the width of the node that the label rescales to. Defaults to 0. 9.

label.norm A single string that is used to normalize label size. If the width of the label is lower than the width of the hypothetical label given by this argument the width of label given by this argument is used instead. Defaults to "OOO" so that every label up to three characters has the same fontsize.

label.scale Logical indicating if labels should be scaled to fit the node. Defaults to TRUE.

edge.labels If FALSE, no edge labels are plotted. If TRUE, numerical edge weights are printed on the edges. This can also be a vector with a label for each edge. Defaults to FALSE. If a label contains an asterisk (e.g. "y1*") then the asterisk will be omitted and the label will be printed in symbol font (use this for Greek letters). Can also be a list with a label as each element, which can be expressions for more advanced mathematical annotation.

edge.label.cex Either a single number or a number per edge used as a scalar of the edge label size. Defaults to 1.

edge.label.bg Either a logical or character vector/matrix. Indicates the background behind edge labels. If TRUE (default) a white background is plotted behind each edge label. If FALSE no background is plotted behind edge labels. Can also be a single color character, a vector or matrix of color vectors for each edge.

bg If this is TRUE, a background is plotted in which node colors cast a light of that color on a black background. Can also be a character containing the color of the background Defaults to FALSE

bgcontrol The higher this is, the less light each node gives if bg=TRUE. Defaults to 6.

bgres square root of the number of pixels used in bg=TRUE, defaults to 100.

trans In weighted graphs: logical indicating if the edges should fade to white (FALSE) or become more transparent (TRUE; use this only if you use a background). In directed graphs this is a value between 0 and 1 indicating the level of transparency. (also used as 'transparency')

fade if TRUE (default) and if 'edge.color' is assigned, transparency will be added to edges that are not transparent (or for which no transparency has been assigned) relative to the edge strength, similar if 'trans' is set to TRUE.

posCol Color of positive edges. Can be a vector of two to indicate color of edges under 'cut' value and color of edges over 'cut' value. If 'fade is set to TRUE the first color will be faded the weaker the edge weight is. If this is only one element this color will also be used for edges stronger than the 'cut' value. Defaults to c("#009900","darkgreen")

negCol Color of negative edges. Can be a vector of two to indicate color of edges under 'cut' value and color of edges over 'cut' value. If 'fade is set to TRUE the first color will be faded the weaker the edge weight is. If this is only one element this color will also be used for edges stronger than the 'cut' value. Defaults to c("#BF0000","red")

unCol Color to indicate the default edge color of unweighted graphs. Defaults to "#808080".

colFactor Exponent of transformation in color intensity of relative strength. Defaults to 1 for linear behavior.

label.color Character containing the color of the labels, defaults to "black"

diag Should the diagonal also be plotted as edges? defaults to FALSE. Can also be "col" to plot diagonal values as vertex colors.

loop If diag=T, this can be used to scale the size of the loop. defaults to 1.

lty Line type, see 'par'

pty See 'par'

borders Logical indicating if borders should be plotted, defaults to TRUE.

border.color Color vector indicating colors of the borders. Is repeated if length is equal to 1. Defaults to "black"

border.width Controls the width of the border. Defaults to 2 and is comparable to 'lwd' argument in 'points'.

shape A character containing the shape of the nodes. `"circle"`, `"square"`, `"triangle"` and `"diamond"` are supported. In addition, can be a name of an element of `polygonList` to plot the corresponding polygon (not reccomended for large graphs), which by default includes shapes `"ellipse"` and `"heart"` Can also be a vector with a shape for each node. Defaults to "circle".

polygonList A list contaning named lists for each element to include polygons to lookup in the `shape` argument. Each element must be named as they are used in `shape` and contain a list with elements `x` and `y` contaning the coordinates of the polygon. By default `ellipse` and `heart` are added to this list. These polygons are scaled according to `vsize` and `vsize2`

gray Logical, set to TRUE to plot the graph in grayscale colors

tooltips A vector with tooltips for each node, only used when filetype='svg' or filetype='tex'

vTrans Transparency of the nodes, must be an integer between 0 and 255, 255 indicating no transparency. Defaults to 255

overlay Logical, should a Venn-diagram like overlay be plotted? If TRUE then for each group a x% confidence region is plotted for the X and Y position, using `ellipse`

overlaySize Specifies the size of the overlay ellipses. Corresponds to the confidence level (default is 0.5)

rotation A vector that can be used to rotate the circles created with the circular layout. Must contain the rotation in radian for each group of nodes. Defaults to zero for each group.

layout.par A list of arguments passed to `qgraph.layout.fruchtermanreingold` when layout="spring" or to an igraph function when such a function is assigned to 'layout'

layout.control A scalar on the size of the circles created with the circular layout.

aspect Should the original aspect ratio be maintained if rescale = TRUE? Defaults to FALSE. Set this to TRUE to keep the aspect ratio of the original layout (e.g. result from layout="spring").

subplots A list with as elements R expressions or NULL for each node. If it is an R expression it is evaluated to create a plot for the node.

images A character vector of length 1 or the same length as the number of nodes indicating the file location of PNG or JPEG images to use as nodes. Can be NA to not plot an image as node and overwrites 'subplots'

Arguments to control edge curvature

curve A value indicating how strongly edges should be curved. Either a single value, a vector (edgelist input) with a value for each edge or a matrix (weights matrix input). NA indicates default curve behavior should be used, which only curves edges if there are multiple edges between two nodes.

curveAll Logical, indicating if all edges should be curved with the value of the 'curve' or only edges between nodes that have share multiple edges.

curveDefault The default curvature. Defaults to 1.

curveShape The shape of the curve, as used in `xspline`. Defaults to `-1`.

curveScale Logical, should curve scale with distance between nodes. Defaults to TRUE. If FALSE, the curve can be exactly determined. Recommended to set to TRUE for graphs and FALSE for diagrams.

curvePivot Quantile to pivot curves on. This can be used to, rather than round edges, make straight edges as curves with "knicks" in them. Can be logical or numeric. `FALSE` (default) indicates no pivoting in the curved edges, a number indicates the quantile (and one minus this value as quantile) on which to pivot curved edges and `TRUE` indicates a value of 0.1.

curvePivotShape The shape of the curve around the pivots, as used in `xspline`. Defaults to `0.25`.

Additional arguments for directed graphs

directed Logical indicating if edges are directed or not. Can be TRUE or FALSE to indicate if all edges are directed, a logical vector (when using edgelists) or a logical matrix (when using weights matrix)

arrows A logical indicating if arrows should be drawn, or a number indicating how much arrows should be drawn on each edge. If this is TRUE, a simple arrow is plotted, if this is a number, arrows are put in the middle of the edges.

arrowAngle Angle of the arrowhead, in radians. Defaults to pi/8 for unweighted graphs and pi/4 for weighted graphs.

asize Size of the arrowhead.

open Logical indicating if open (TRUE) or closed (FALSE) arrowheads should be drawn.

bidirectional If this is TRUE, Then directional edges between nodes that have two edges between them are not curved. Defaults to FALSE. Can also be a logical vector (when using edgelists) or a logical matrix (when using weights matrix)

Arguments for graphs based on significance values

mode This argument defines the mode used for coloring the edges. The default, "strength" assumes each edge weight indicates the strength of connection centered around and makes positive

edges green and negative edges red. If this is set to "sig" then the edge weights are assumed to be significance values and colored accordingly. This can also include negative values, which will be interpreted as p-values based on negative statistics.

alpha A vector of max 4 elements indicating the alpha level cutoffs. Defaults to c(0.0001,0.001,0.01,0.05)

OmitInsig Logical indicating if edge weights with a p-value over the highest alpha level should be omitted. Defaults to FALSE, can be used with any mode

sigScale The function used to scale the edges if mode="sig". Defaults to $function(x)0.8*(1-x)^(log(0.4/0.8,1-0.05))$

bonf Logical indicating if a bonferonni correction should be applied if mode="sig". If so p-values are multiplied by the number of edges

Additional options for Correlation Matrices

graph Type of graph to be made, for use with a correlation matrix as input. "association" will plot the matrix as is, "concentration" will first compute partial correlations between each pair of nodes (controlled for all other variables) and "factorial" will create a graph based on an exploratory factor analysis. Finally "sig" will transform all correlations in p-values (using the fdrtool package; Korbinian Strimmer, 2009) and force mode="sig". "sig2" will do the same but show p-values based on negative statistics in shades of orange

Arguments for plotting scores on nodes

scores This argument can be used to plot scores of an individual on the test. Should be a vector with the scores for each item. Currently this can only be integer values (e.g.\ LIKERT scales).

scores.range Vector of length two indicating the range of the scores, if scores is assigned.

Arguments for manually defining graphs

mode The mode argument (see section on significance graph arguments) can also be used to make the weights matrix correspond directly to the width of the edges (as in lwd of plot()). To do this, set mode to "direct".

edge.color This argument can be used to overwrite the colors. Can be either a single value to make all edges the same color, a matrix with a color for each edge (when using a weights matrix) or a vector with a color for each edge (when using an edgelist). NA indicates that the default color should be used.

Arguments for knots (tying together edges)

knots This argument can be used to tie edges together in their center, which can be useful to, for example, indicate interaction effects. This argument can be assigned a list where each element is a vector containing the edge numbers that should be knotted together. Another option is to assign the argument a integer vector (for edgelists) or a matrix (for weight matrices) with 0 indicating edges that should not be tied together, and increasing numbers indicating each knot.

knot.size The size of the knots. Can be of length one or a vector with the size of each knot. Similar to 'vsize'. Defaults to 1.

knot.color The color of the knots. Can be of length one or a vector with the size of each knot. Defaults to NA, which will result in a mix of the knotted edge colors.

knot.borders Logical indicating if a border should be plotted around the knot. Can be of length one or a vector with the size of each knot. Works similar to 'borders'. Defaults to FALSE

knot.border.color Color of the knot borders. Can be of length one or a vector with the size of each knot. Works similar to 'border.color'. Defaults to "black"

knot.border.width Width of the knot borders. Can be of length one or a vector with the size of each knot. Works similar to 'border.width'. Defaults to 1

Arguments for bars

bars A list with for each node containing either NULL or a vector with values between 0 and 1 indicating where bars should be placed inside the node.

barSide Integer for each node indicating at which side the bars should be drawn. 1, 2, 3 or 4 indicating at bottom, left, top or right respectively.

barColor A vector with for each node indicating the color of bars. Defaults to the border color of the node.

barLength A Vector indicating the relative length of bars of each node compared to the node size. Defaults to 0.5.

barsAtSide Logical, should bars be drawn at the side of a node or at its center? Defaults to FALSE.

Additional arguments

edgelist Logical, if TRUE 'input' is assumed to be an edgelist, else if FALSE input is assumed to be a weights matrix. By default this is chosen automatically based on the dimensions of 'input' and this argument is only needed if the dimensions are ambiguous (square matrix with 2 or 3 rows/columns)

weighted Logical that can be used to force either a weighted graph (TRUE) or an unweighted graph(FALSE).

nNodes The number of nodes, only needs to be specified if the first argument is an edge-list and some nodes have no edges

DoNotPlot Runs qgraph but does not plot. Useful for saving the output (i.e. layout) without plotting

plot Logical. Should a new plot be made? Defaults to TRUE. Set to FALSE to add the graph to the existing plot.

rescale Logical. Defines if the layout should be rescaled to fit the -1 to 1 x and y area. Defaults to TRUE. Can best be used in combination with plot=FALSE.

layoutScale A vector with a scalar for respectively the x and y coordinates of the layout (which default plotting area is from -1 to 1 on both x and y axis). Setting this to e.g. c(2,2) would make the plot twice as big. Use this in combination with 'layoutOffset' and 'plot' arguments to define the graph placement on an existing plot.

layoutOffset A vector with the offset to the x and coordinates of the center of the graph (defaults to (0,0)). Use this in combination with 'layoutScale' and 'plot' arguments to define the graph placement on an existing plot.

Using qgraph to plot graphs

The first argument of qgraph(), 'input', is the input. This can be a number of objects but is mainly either a weights matrix or an edgelist. Here we will assume a graph is made of n nodes connected by m edges. qgraph is mainly aimed at visualizing (statistical) relationships between variables as weighted edges. In these edge weights a zero indicates no connection and negative values are comparable in strength to positive values. Many (standardized) statistics follow these rules, the most important example being correlations. In the special case where all edge weights are either 0 or 1 the weights matrix is interpreted as an adjacency matrix and an unweighted graph is made.

a weights matrix is a square n by n matrix in which each row and column represents a node. The element at row i and column j indicates the connection from node i to node j. If the weights matrix is symmetrical an undirected graph is made and if the matrix is asymmetrical a directed graph is made (note that due to floating point errors seemingly symmetrical matrices may actually be asymmetrical). These matrices occur naturally in statistics or can easily be obtained through matrix algebra, the most important example being correlation matrices.

An edgelist can also be used. This is a m by 2 matrix (not a list!) in which each row indicates an edge. The first column indicates the number of the start of the edge and the second column indicates the number of the end of the edge. The number of each node is a unique integer between 1 and n. The total number of nodes will be estimated by taking the highest value of the edgelist. If this is incorrect (there are nodes with no edges beyond the ones already specified) the 'nNodes' argument can be used. If an integer between 1 and n is missing in the edgelist it is assumed to be a node with no edges. To create a weighted graph edge weights can be added as a third column in the edgelist. By default using an edgelist creates a directed graph, but this can be set with the 'directed' argument.

Interpreting graphs

In weighted graphs green edges indicate positive weights and red edges indicate negative weights. The color saturation and the width of the edges corresponds to the absolute weight and scale relative to the strongest weight in the graph. It is possible to set this strongest edge by using the 'maximum' argument. When 'maximum' is set to a value above any absolute weight in the graph that value is considered the strongest edge (this must be done to compare different graphs; a good value for correlations is 1). Edges with an absolute value under the 'minimum' argument are omitted (useful to keep filesizes from inflating in very large graphs).

In larger graphs the above edge settings can become hard to interpret. With the 'cut' argument a cutoff value can be set which splits scaling of color and width. This makes the graphs much easier to interpret as you can see important edges and general trends in the same picture. Edges with absolute weights under the cutoff score will have the smallest width and become more colorful as they approach the cutoff score, and edges with absolute weights over the cutoff score will be full red or green and become wider the stronger they are.

Specifying the layout

The placement of the nodes (i.e. the layout) is specified with the 'layout' argument. It can be manually specified by entering a matrix for this argument. The matrix must have a row for each node and two columns indicating its X and Y coordinate respectively. qgraph plots the nodes on a (-1:1)(-1:1) plane, and the given coordinates will be rescaled to fit this plane unless 'rescale' is FALSE (not recommended). Another option to manually specify the layout is by entering a matrix

with more then two columns. This matrix must then consist of zeroes and a number (the order in the weights matrix) for each node indicating it's place. For example:

0 0 2 0 0

1 0 3 0 4

will place node 2 at the top in the center, node 1 at the bottom left corner, node 3 at the bottom in the center and node 4 at the bottom right corner. It is recommended however that one of the integrated layouts is used. 'layout' can be given a character as argument to accomplish that. layout="circular" will simply place all nodes in a circle if the groups argument is not used and in separate circles per group if the groups argument is used (see next section).

The circular layout is convenient to see how well the data conforms to a model, but to show how the data clusters another layout is more appropriate. By specifying layout="spring" the Fruchterman-reingold algorithm (Fruchterman & Reingold, 1991), which has been ported from the SNA package (Butts, 2010), can be used to create a force-directed layout. In principle, what this function does is that each node (connected and unconnected) repulse each other, and connected nodes also attract each other. Then after a number of iterations (500 by default) in which the maximum displacement of each node becomes smaller a layout is achieved in which the distance between nodes correspond very well to the absolute edge weight between those nodes.

A solution to use this function for weighted graphs has been taken from the igraph package (Csardi G & Nepusz T, 2006) in which the same function was ported from the SNA package. New in qgraph are the option to include constraints on the nodes by fixing a coordinate for nodes or reducing the maximum allowed displacement per node. This can be done with the 'layout.par' argument. For more information see `qgraph.layout.fruchtermanreingold`.

By default, 'layout' is set to "spring" for unweighted and directed graphs and "circular" otherwise.

Specifying a measurement model

A measurement model can be specified with the 'groups' argument. This must be a list in which each element is a vector containing the numbers of nodes that belong together (numbers are taken from the order in the weights matrix). All numbers must be included. If a groups list is specified the "groups" layout can be used to place these nodes together, the nodes in each group will be given a color, and a legend can be plotted (by setting 'legend' to TRUE). The colors will be taken from the 'color' argument, or be generated with the `rainbow` function.

Output

By default qgraph will plot the graph in a new R window. However the graphs are optimized to be plotted in a PDF file. To easily create a pdf file set the 'filetype' argument to "pdf". 'filename' can be used to specify the filename and folder to output in. 'height' and 'width' can be used to specify the height and width of the image in inches. By default a new R window is opened if the current device is the NULL-device, otherwise the current device is used (note that when doing this 'width' and 'height' still optimize the image for those widths and heights, even though the output screen size isn't affected, this is especially important for directed graphs!).

Furthermore filetype can also be set to numerous other values. Alternatively any output device in R can be used by simply opening the device before calling qgraph and closing it with dev.off() after calling qgraph.

The graphs can also be outputted in an SVG file using the RSVGTipsDevice package (Plate, 2009). An SVG image can be opened in most browsers (firefox and chrome are recommended), and can be

used to display tooltips. Each node can be given a tooltip with the 'tooltips' argument. The function qgraph.svg can be used to make a battery of svg pictures with hyperlinks to each other, working like a navigation menu.

IMPORTANT NOTE: RSVGTipsDevice is a 32-bit only package, so SVG functionality is not available in 64bit versions of R.

Finally, the filetype 'tex' can be used. This uses the tikzDevice package to create a LaTeX file that can then be compiled in your LaTeX compiler to create a pdf file. The main benefit of this over plotting directly in a pdf file is that tooltips can be added which can be viewed in several PDF document readers (Adobe Reader is recommended for the best result).

Manual specification of color and width

In qgraph the widths and colors of each edge can also be manually controlled. To directly specify the width of each edge set the 'mode" argument to "direct". This will then use the absolute edge weights as the width of each edge (negative values can still be used to make red edges). To manually set the color of each edge, set the 'edge.color' argument to a matrix with colors for each edge (when using a weights matrix) or a vector with a color for each edge (when using an edgelist).

Additional information

By default, edges will be straight between two nodes unless there are two edges between two nodes. To overwrite this the 'bidirectional' argument can be set to TRUE, which will turn two edges between two nodes into one bidirectional edge. 'bidirectional' can also be a vector with TRUE or FALSE for each edge.

To specify the strength of the curve the argument 'curve' can be used (but only in directional graphs). 'curve' must be given a numerical value that represent an offset from the middle of the straight edge through where the curved edge must be drawn. 0 indicates no curve, and any other value indicates a curve of that strength. A value of 0.3 is recommended for nice curves. This can be either one number or a vector with the curve of each edge.

Nodes and edges can be given labels with the 'labels' and the 'edge.labels' arguments. 'labels' can be set to FALSE to omit labels, TRUE (default) to set labels equal to the node number (order in the weights matrix) or it can be a vector with the label for each node. Edge labels can also be set to FALSE to be omitted (default). If 'edge.labels' is TRUE then the weight of each label is printed. Finally, 'edge.labels' can also be a vector with the label for each edge. If a label (both for edges and nodes) contain an asterisk then the asterisk is omitted and that label is printed in the symbol font (useful to print Greek letters).

A final two things to try: the 'scores' argument can be given a vector with the scores of a person on each variable, which will then be shown using colors of the nodes, And the 'bg' argument can be used to change the background of the graph to another color, or use bg=TRUE for a special background (do set transparency=TRUE when using background colors other then white).

Debugging

If this function crashes for any reason with the filetype argument specified, run:

dev.off()

To shut down the output device!

Author(s)

Sacha Epskamp <qgraph@sachaepskamp.com>

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Carter T. Butts <buttsc@uci.edu> (2010). sna: Tools for Social Network Analysis. R package version 2.2-0. http://CRAN.R-project.org/package=sna

Csardi G, Nepusz T (2006). The igraph software package for complex network research, InterJournal, Complex Systems 1695. http://igraph.sf.net

Korbinian Strimmer (2009). fdrtool: Estimation and Control of (Local) False Discovery Rates. R package version 1.2.6. http://CRAN.R-project.org/package=fdrtool

Plate, T. <tplate@acm.org> and based on RSvgDevice by T Jake Luciani <jakeluciani@yahoo.com> (2009). RSVGTipsDevice: An R SVG graphics device with dynamic tips and hyperlinks. R package version 1.0-1.

Fruchterman, T. & Reingold, E. (1991). Graph drawing by force-directed placement. Software - Pract. Exp. 21, 1129?1164.

See Also

qgraph qgraph.gui qgraph.animate qgraph.efa qgraph.pca qgraph.loadings qgraph.sem qgraph.lavaan qgraph.cfa qgraph.svg qgraph.panel

Examples

```
### BIG 5 DATASET ###
# Load big5 dataset:
data(big5)
data(big5groups)

# Correlations:
Q <- qgraph(cor(big5),minimum=0.25,cut=0.4,vsize=1.5,groups=big5groups,
  legend=TRUE,borders=FALSE)
title("Big 5 correlations",line=2.5)

## Not run:
# Same graph with spring layout:
Q <- qgraph(Q,layout="spring")
title("Big 5 correlations",line=2.5)

# Same graph with Venn diagram overlay:
qgraph(Q,overlay=TRUE)
title("Big 5 correlations",line=2.5)

# Same graph with different color scheme:
qgraph(Q,posCol="blue",negCol="purple")
title("Big 5 correlations",line=2.5)
```

```
# Significance graph (circular):
qgraph(Q,graph="sig",layout="circular")
title("Big 5 correlations (p-values)",line=2.5)

# Significance graph:
qgraph(Q,graph="sig")
title("Big 5 correlations (p-values)",line=2.5)

# Significance graph (distinguishing positive and negative statistics):
qgraph(Q,graph="sig2")
title("Big 5 correlations (p-values)",line=2.5)

# Grayscale graphs:
qgraph(Q,gray=TRUE,layout="circular")
title("Big 5 correlations",line=2.5)

qgraph(Q,graph="sig",gray=TRUE)
title("Big 5 correlations (p-values)",line=2.5)

# Correlations graph with scores of random subject:
qgraph(cor(big5),minimum=0.25,cut=0.4,vsize=1.5,groups=big5groups,legend=TRUE,
  borders=FALSE,scores=as.integer(big5[sample(1:500,1),]),scores.range=c(1,5))
title("Test scores of random subject",line=2.5)
layout(1)

# EFA:
big5efa <- factanal(big5,factors=5,rotation="promax",scores="regression")
qgraph(big5efa,groups=big5groups,layout="circle",rotation="promax",minimum=0.2,
  cut=0.4,vsize=c(1.5,15),borders=FALSE,vTrans=200)
title("Big 5 EFA",line=2.5)

# PCA:
library("psych")
big5pca <- principal(cor(big5),5,rotate="promax")
qgraph(big5pca,groups=big5groups,layout="circle",rotation="promax",minimum=0.2,
  cut=0.4,vsize=c(1.5,15),borders=FALSE,vTrans=200)
title("Big 5 PCA",line=2.5)

#### UNWEIGHTED DIRECTED GRAPHS ###
set.seed(1)
adj=matrix(sample(0:1,10^2,TRUE,prob=c(0.8,0.2)),nrow=10,ncol=10)
qgraph(adj)
title("Unweighted and directed graphs",line=2.5)

# Save plot to nonsquare pdf file:
qgraph(adj,filetype='pdf',height=5,width=10)


#### EXAMPLES FOR EDGES UNDER DIFFERENT ARGUMENTS ###
# Create edgelist:
dat.3 <- matrix(c(1:15*2-1,1:15*2),,2)
dat.3 <- cbind(dat.3,round(seq(-0.7,0.7,length=15),1))
```

```
# Create grid layout:
L.3 <- matrix(1:30,nrow=2)

# Different esize:
qgraph(dat.3,layout=L.3,directed=FALSE,edge.labels=TRUE,esize=14)

# Different esize, strongest edges omitted (note how 0.4 edge is now
# just as wide as 0.7 edge in previous graph):
qgraph(dat.3[-c(1:3,13:15),],layout=L.3,nNodes=30,directed=FALSE,
       edge.labels=TRUE,esize=14)

# Different esize, with maximum:
qgraph(dat.3,layout=L.3,directed=FALSE,edge.labels=TRUE,esize=14,maximum=1)
title("maximum=1",line=2.5)

qgraph(dat.3[-c(1:3,13:15),],layout=L.3,nNodes=30,directed=FALSE,edge.labels=TRUE,
       esize=14,maximum=1)
title("maximum=1",line=2.5)

# Different minimum
qgraph(dat.3,layout=L.3,directed=FALSE,edge.labels=TRUE,esize=14,minimum=0.1)
title("minimum=0.1",line=2.5)

# With cutoff score:
qgraph(dat.3,layout=L.3,directed=FALSE,edge.labels=TRUE,esize=14,cut=0.4)
title("cut=0.4",line=2.5)

# With details:
qgraph(dat.3,layout=L.3,directed=FALSE,edge.labels=TRUE,esize=14,minimum=0.1,
  maximum=1,cut=0.4,details=TRUE)
title("details=TRUE",line=2.5)

### Additional topics ###

# Trivial example of manually specifying edge color and widths:
E <- as.matrix(data.frame(from=rep(1:3,each=3),to=rep(1:3,3),width=1:9))
qgraph(E,mode="direct",edge.color=rainbow(9))

## pcalg support
# Example from pcalg vignette:
library("pcalg")
data(gmI)
suffStat <- list(C = cor(gmI$x), n = nrow(gmI$x))
pc.fit <- pc(suffStat, indepTest=gaussCItest,
             p = ncol(gmI$x), alpha = 0.01)

qgraph(pc.fit)

## End(Not run)
```

qgraph.animate *Animate a growing network*

Description

This function is meant to facilitate the creation of animations based on growing networks. Networks are created based on the Fruchterman Reingold algorithm, which is constraint by limiting the maximum displacement of nodes that are already in the graph.

Usage

```
qgraph.animate(input, ind = NULL, ..., constraint = 10, growth = "order",
  titles = NULL, sleep = 0)
```

Arguments

input	A weights matrix of the graph or a list of weigths matrices with different weights of the same graph (see details). See qgraph. Edgelists are currently not supported.
ind	An object that specifies which nodes ar included or excluded. See details.
...	Additional arguments sent to qgraph
constraint	The constraint factor of included nodes. See details. Defaults to 10 for an soft-constrained animation. Set to Inf for a hard-constrained animation.
growth	The way nodes are added by default. Set to "order" to include nodes in the order they appear in the weigths matrix and to "degree" to include nodes based on their degree (high degree first)
titles	Optional vector with a title for each plot
sleep	Optional value sent to Sys.sleep() for showing the animation in R

Details

Let n be the number of nodes in total in the graph.

This function is designed to facilitate the production of animations by constraining the Fruchterman Reingold algorithm. Several frames are plotted of (a subset of) the same graph. If a node was already in the graph its maximum displacement per iteration of Fruchterman Reingold is equal to the number of nodes times the inverse of the constraint argument (so by default n/10). The higher this constraint value the stricter nodes stay in the same place between plots.

How many and which plots are made are defined by the 'input' and 'ind' arguments. There are two ways to specify the 'input' argument, either by speficying one weigths matrix or by specifying a list of weights matrices. In the sections below is explained what both of these methods do and how they are used.

This function, since it can be seen as an expression that makes several plots, works well in combination with the animation package for saving the animation to a wide variety of filetypes.

Single weigths matrix

If 'input' is a single weigths matrix then in each frame a subset of the same graph is plotted. This is especially usefull for animating the growth of a network. Which nodes are in each frame is determined by the 'ind' argument.

If 'int' is not specified an animation is created in which in each frame a single node is added. This node is either in order of apearance in the weigths matrix or by its degree, which is determined with the 'growth' argument.

If 'ind' is a logical vector of length n than the first frame will contain the nodes specified with this vector and all other frames will grow in the same way as explained above (each step one node is added).

If 'ind' is a numeric vector of length n which contains all integers between 1 and n (a single entry per node) then the first frame starts with only the node specified in the first element of the vector and in frame i the ith element is added (each step one node is added).

If 'ind' is a list with numeric vectors as elements containing integers between 1 and n then in frame i the nodes from the ith element of the list will be added. Node numbers that occur multiple times in the list are ignored (they are already added the first time).

Finally, if 'ind' is a logical matrix with n columns and an arbitrary amount of rows, then in frame i only the nodes that are TRUE in row i are included. This is the only way to specify removal of nodes.

List of weigths matrices

The 'input' argument can also be given a list of weigths matrices if all these matrices have the same dimension (i.e.\ only the weights differ). If this is done than in frame i the ith weigths matrix is used. This is especially usefull for animating the change in a graph.

In this case, the 'ind' argument behaves differently. If this argument is not specified then in each frame all nodes are included.

If 'ind' is a logical vector of length n then only one plot is made with the nodes specified with that vector, and only if the length of 'input' is one.

Other methods woth in the same way as above. However, if the 'ind' argument indicates a different number of frames than the 'input' argument the function will stop and give an error.

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

See Also

qgraph

Examples

```
## Not run:

## For these examples, first generate a scale free network using preferential attachment:

# Number of nodes:
n <- 100
# Empty vector with Degrees:
Degs <- rep(0, n)
# Empty Edgelist:
E <- matrix(NA, n - 1, 2)
# Add and connect nodes 1 and 2:
E[1, ] <- 1:2
Degs[1:2] <- 1
# For each node, add it with probability proportional to degree:
for (i in 2:(n - 1))
{
E[i, 2] <- i + 1
con <- sample(1:i, 1, prob = Degs[1:i]/sum(Degs[1:i]),i)
Degs[c(con,i+1)] <- Degs[c(con,i+1)] + 1
E[i, 1] <- con
}

# Because this is an edgelist we need a function to convert this to an adjacency matrix:
E2adj <- function(E,n)
{
  adj <- matrix(0,n,n)
  for (i in 1:nrow(E))
  {
      adj[E[i,1],E[i,2]]  <- 1
  }
  adj <- adj + t(adj)
  return(adj)
}



### EXAMPLE 1: Animation of construction algorithm: ###
adjs <- lapply(1:nrow(E),function(i) E2adj(E[1:i,,drop=FALSE],n))
qgraph.animate(adjs,color="black",labels=FALSE,sleep=0.1)
rm(adjs)

### EXAMPLE 2: Add nodes by final degree: ###
adj <- E2adj(E,n)
qgraph.animate(E2adj(E,n),color="black",labels=FALSE,constraint=100,sleep=0.1)

### EXAMPLE 3: Changing edge weights: ###
adjW <- adj*rnorm(n^2)
adjW <- (adjW + t(adjW))/2
adjs <- list(adjW)
for (i in 2:100)
{
```

```
  adjW <- adj*rnorm(n^2)
  adjW <- (adjW + t(adjW))/2
  adjs[[i]] <- adjs[[i-1]] + adjW
}
qgraph.animate(adjs,color="black",labels=FALSE,constraint=100,sleep=0.1)


## End(Not run)
```

---

qgraph.cfa *CFA using Structural Equation Modelling*

Description

This function performs a simple confirmatory factor analysis using sem (Fox, 2010) or lavaan (Rosseel, 2011).

Usage

```
qgraph.cfa(S, N, groups=NULL, ..., pkg = "sem", labels=NULL, fun = qgraph, opts = list())
```

Arguments

S	A covariance matrix
N	The number of observations
groups	The groups list, see qgraph. This must be a list in which each element is a factor indicating which variables load on the same factor
...	Arguments passed to 'fun'
pkg	A string indicating which package should be used for estimating the model. Currently "lavaan" and "sem" (default) are supported
labels	A vector indicating the label of each variable
fun	A function to which the results are send. Defaults to qgraph, but can be any function that can handle the output (e.g. qgraph.lavaan, qgraph.sem, summary, print). If pkg="sem" then this can also be qgraph.loadings.
opts	A list containing arguments that are sent to either sem or cfa

Details

This function can be used to perform a simple confirmatory factor analasys using regular qgraph input. The function computes a model and then sends it to sem (sem; Fox, 2010) or cfa (lavaan; Rosseel, 2011). based on the package used either a "sem" object or a "lavaan" object is returned that can be used for manual inspection or to sent to qgraph.sem or qgraph.lavaan.

The model that is estimated is a first order factor model in which each variable loads on one factor and the factors are correlated. This model is specified with the 'groups' argument. This must be a list in which each element represents a factor. Each element of the list must be a vector indicating

which variables load on the same factor. The model is identified by fixing the first loading of each factor to 1, which should be an identifying restriction if there are at least 4 variables per factor.

The function also sends its results to another function for visualization. If this is qgraph, the default, then a visualization of the standardized coeficients is plotted.

Currently the sem package is better supported in qgraph, but this will change in a future version. Using the lavaan package can greatly reduce computation time.

Value

A "sem" object, see sem

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

John Fox with contributions from Adam Kramer <jfox@mcmaster.ca> and Michael Friendly (2010). sem: Structural Equation Models. R package version 0.9-21. http://CRAN.R-project.org/package=sem

See Also

qgraph qgraph.sem qgraph.lavaan qgraph.loadings qgraph.semModel sem

Examples

```
## Not run:
# Simulate dataset:
set.seed(2)
eta<-matrix(rnorm(200*5),ncol=5)
lam<-matrix(rnorm(50*5,0,0.15),50,5)
lam[apply(diag(5)==1,1,rep,each=10)]<-rnorm(50,0.7,0.3)
th<-matrix(rnorm(200*50),ncol=50)
Y<-eta%*%t(lam)+th

# Create groupslist
gr<-list(1:10,11:20,21:30,31:40,41:50)

# Using "lavaan" package:
res <- qgraph.cfa(cov(Y),N=200,groups=gr,pkg="lavaan",vsize.man=2,vsize.lat=10)

qgraph.lavaan(res,filename="lavaan",legend=FALSE,groups=gr,edge.label.cex=0.6)

# Using "sem" package:
res <- qgraph.cfa(cov(Y),N=200,groups=gr,pkg="sem",vsize.man=2,vsize.lat=10,fun=qgraph.loadings)

qgraph.semModel(res,edge.label.cex=0.6)
```

```
qgraph(res,edge.label.cex=0.6)

qgraph.sem(res,filename="sem",legend=FALSE,groups=gr,edge.label.cex=0.6)

### Big 5 dataset ###
data(big5)
data(big5groups)

fit <- qgraph.cfa(cov(big5),nrow(big5),big5groups,pkg="lavaan",opts=list(se="none"),
vsize.man=1,vsize.lat=6,edge.label.cex=0.5)
print(fit)

## End(Not run)
```

---

qgraph.efa *qgraph.efa*

Description

This function performs an Exploratory Factor Analysis (EFA) using the factanal (stats) function and sends the acquired factor loadings to qgraph.loadings.

Usage

```
qgraph.efa(dat,factors=1,...,rotation="promax",residuals=TRUE,
factorCors=NULL,scores="regression",
corMat=nrow(dat)==ncol(dat) && all(dat==t(dat)))
```

Arguments

dat	A correlation matrix, data matrix or a "factanal" object
factors	The number of factors to extract
rotation	rotation to be used. Can be "varimax", "promax" or "none"
residuals	Logical indicating if residuals should be plotted. Defaults to TRUE
factorCors	Logical indicating if correlations of factors should be extracted and plotted. Defaults to FALSE if a correlation matrix is used and TRUE if a data matrix is used.
...	arguments passed to qgraph.loadings
scores	Method used to extract scores in factanal
corMat	Logical indicating if the 'dat' object is a correlation matrix (TRUE) or data matrix (FALSE)

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

See Also

qgraph qgraph.pca qgraph.loadings

Examples

```
## Not run:
data(big5)
data(big5groups)

qgraph.efa(big5,5,groups=big5groups,rotation="promax",minimum=0.2,cut=0.4,
   vsize=c(1,7),borders=FALSE,vTrans=200)

# Tree layout:
qgraph.efa(big5,5,groups=big5groups,rotation="promax",minimum=0.2,cut=0.4,
   vsize=c(1,7),borders=FALSE,layout="tree",width=20,filetype="R")

## End(Not run)
```

---

qgraph.gui *Graphical user interface for qgraph*

Description

This function works the same as qgraph except that a graphical user interface is opened allowing the user to change parameters.

Usage

```
qgraph.gui(input, corMat, ...)
```

Arguments

input	Input, see qgraph
corMat	Logical, is 'input' a correlation or covariance matrix. If this is TRUE a different GUI window is opened with more options. By default this is TRUE only if 'input' is a symmetrical matrix with all absolute values less or equal than 1 and all diagonal elements equal to 1. If a covariance matrix is used for 'input' and this argument is set to TRUE the plots are made using a 'cov2cor' call, but the covariance matrix is used in computing EFA and PCA results.
...	Arguments passed to qgraph

Details

Press enter after entering values to save them for the next plot.

Author(s)

Sacha Epskamp <qgraph@sachaepskamp.com>

References

Adrian Bowman, Ewan Crawford, Gavin Alexander, Richard W. Bowman (2007). rpanel: Simple Interactive Controls for R Functions Using the tcltk Package. Journal of Statistical Software, 17(9), 1-18. URL http://www.jstatsoft.org/v17/i09/.

See Also

qgraph

Examples

```
## Not run:
data(big5)
data(big5groups)

qgraph.gui(cor(big5),groups=big5groups)

## End(Not run)
```

`qgraph.lavaan` *qgraph: Structural Equation Modelling*

Description

This functions uses a "lavaan" object from the lavaan package (Rosseel, 2011) and outputs a multi-page pdf file containing path diagram, graphs of the parameter estimates and graphs of the implied and observed covariance and correlation matrices.

Usage

```
qgraph.lavaan(
fit,
...,
layout="circle",
groups=NULL,
vsize.man=3,
vsize.lat=6,
filename="qgraph",
filetype="pdf",
residuals=TRUE,
```

```
include=1:12,
curve=0,
residSize=0.2,
onefile=TRUE,
width=12,
height=8,
titles=TRUE)
```

Arguments

fit	A "lavaan" object containing the fit of a SEM model (obtained from e.g. sem and cfa)
...	arguments passed to qgraph. This is both for the path diagram and for the correlation/covariance plots.
layout	The layout used for the path diagram. Can be "tree", "spring", "circle" and "springtree"
groups	An optional list containing the measurement model, see qgraph
vsize.man	Size of the manifest variables in the path diagram
vsize.lat	Size of the latent variables in the path diagram
filename	Name of the file (will be extended with the filetype)
filetype	The filetype to be used. Can be "pdf" to make a pdf (default) or anything else to plot in R. More filetypes will be supported in a future version.
residuals	Omitting residuals is currently not supported for qgraph.lavaan, leave this to TRUE
include	A vector indicating which panels should be included in the output
curve	Numerical value indicating the curve of edges that are on the same level in the "tree" layout, See details. This represent an offset from the middle of the straight edge through where the curved edge must be drawn. 0 indicates no curve, and any other value indicates a curve of that strength. Defaults to 0.2
residSize	Size of the residual arrows
onefile	Logical indicating if a multi-page pdf should be produced. If FALSE each plot will be a new pdf. Use this only with panels=1 and filename="(Arbitrary name)%03d".
width	Width of each panel, in inches
height	Height of each panel, in inches
titles	Logical indicating if titles should be printed

Details

This function uses a "lavaan" object and outputs a multi-page pdf file. The function reads the 'lavaan' object and creates a residual variable for each variable present in the model. Layout options include a tree-layout (layout="tree") in which each variable is placed as a node on one of four vertical levels. At the bottom are the residuals of the manifest variables placed, Above that the manifest variables, above that the latent variables and at the top the residuals of the latent variables.

The nodes are evenly spaced horizontally in order of appearance in the model (residuals are placed at the same horizontal position of their corresponding variable). So the order of specifying in the model defines the order of placement in the path diagram. If the 'residuals' argument is FALSE then residuals are omitted and this diagram will only contain two levels.

Alternatively the 'spring' layout can be used (layout="spring"). This will use the Fruchterman-reingold algorithm (Fruchterman & Reingold, 1991), which has been ported from the 'sna' package (Butts, 2010). This is a force-directed algorithm. It is best to use this in combination with residuals=FALSE. Another option is a circular layout (default), which is the same as the tree except that the levels are placed in inner circles rather than horizontal lines.

Names for variables used in the model specification are passed to the path diagram. To keep the model readable it is advised to limit these names to 3 characters.

Note

This is the first isntallment of qgraph.sem for the Lavaan package. This function will likely be changed a lot in future installments.

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Carter T. Butts <buttsc@uci.edu> (2010). sna: Tools for Social Network Analysis. R package version 2.2-0. http://CRAN.R-project.org/package=sna

Yves Rosseel <Yves.Rosseel@UGent.be> (2011). lavaan: Latent Variable Analysis. R package version 0.4-7. http://CRAN.R-project.org/package=lavaan

Fruchterman, T. & Reingold, E. (1991). Graph drawing by force-directed placement. Software - Pract. Exp. 21, 1129?1164.

See Also

qgraph qgraph.cfa qgraph.sem sem

Examples

```
## Not run:
## The industrialization and Political Democracy Example
# Example from lavaan::sem help file:
require("lavaan")
     ## Bollen (1989), page 332
     model <- '
       # latent variable definitions
          ind60 =~ x1 + x2 + x3
          dem60 =~ y1 + y2 + y3 + y4
          dem65 =~ y5 + equal("dem60=~y2")*y6
```

```
                    + equal("dem60=~y3")*y7
                    + equal("dem60=~y4")*y8

      # regressions
        dem60 ~ ind60
        dem65 ~ ind60 + dem60

      # residual correlations
        y1 ~~ y5
        y2 ~~ y4 + y6
        y3 ~~ y7
        y4 ~~ y8
        y6 ~~ y8
    '

    fit <- sem(model, data=PoliticalDdemocracy)

# Plot standardized model (numerical):
qgraph.lavaan(fit,layout="tree",vsize.man=5,vsize.lat=10,
filetype="",include=4,curve=-0.4,edge.label.cex=0.6)

# Plot standardized model (graphical):
qgraph.lavaan(fit,layout="tree",vsize.man=5,vsize.lat=10,
filetype="",include=8,curve=-0.4,edge.label.cex=0.6)

# Create output document:
qgraph.lavaan(fit,layout="spring",vsize.man=5,vsize.lat=10,
filename="lavaan")


## End(Not run)
```

`qgraph.layout.fruchtermanreingold` *qgraph.layout.fruchtermanreingold*

Description

This is a wrapper for the function that returns the x and y coordinates of the graph based on the Fruchterman Reingold algorithm (Fruchterman & Reingold, 1991), which was ported from the SNA package (Butts, 2010). This function is used in `qgraph` and is not designed to be used separately. See details for using constraints in this layout.

Usage

```
qgraph.layout.fruchtermanreingold(edgelist, weights=NULL, vcount=NULL,
niter=NULL, max.delta=NULL, area=NULL, cool.exp=NULL, repulse.rad=NULL,
init=NULL, groups=NULL, rotation=NULL, layout.control=0.5, constraints=NULL)
```

Arguments

edgelist	A matrix with on each row the nodes at the start and the node at the end of each edge.
weights	A vector containing the edge weights.
vcount	The number of nodes.
niter	Number of iterations, default is 500.
max.delta	Maximum displacement, default is equal to the number of nodes.
area	The area of the plot, default is the square of the number of nodes.
cool.exp	Cooling exponent, default is 1.5.
repulse.rad	Repulse radius, defaults to the cube of the number of nodes.
init	Matrix with two columns and a row for each node containing the initial X and Y positions.
groups	See qgraph
rotation	See qgraph
layout.control	See qgraph
constraints	A constraints matrix with two columns and a row for each node containing a NA if the node is free or a fixed value for one of the coordinates.

Details

All arguments for this function can be passed from qgraph to this function by using the 'layout.par' argument, which must be a list containing the arguments. This can be used to constrain the layout in two ways:

Hard constraints

By using the 'constraints' argument the X and Y positions of each node can be fixed to a certain value. The 'constraint' argument must be given a matrix with two columns and a row for each node. An NA means that that coordinate for that node is free, and a value means it is fixed to that value.

Soft constraints

Soft constraining can be done by varying the 'max.delta' argument. This can be a single number, but also a vector containing the maximum displacement per step for each node. The default value is the number of nodes, so by setting this to a lower value for some nodes the node won't move so much. Use this in combination with the 'init' argument to make sure nodes don't move too much from their initial setup. This can be useful when adding a new node to an existing network and if you don't want the network to completely change.

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Carter T. Butts <buttsc@uci.edu> (2010). sna: Tools for Social Network Analysis. R package version 2.2-0. http://CRAN.R-project.org/package=sna

Fruchterman, T. & Reingold, E. (1991). Graph drawing by force-directed placement. Software - Pract. Exp. 21, 1129<96>1164.

See Also

qgraph

Examples

```
## Not run:
# This example makes a multipage PDF that contains images
# Of a building network using soft constraints.

# Each step one node is added with one edge. The max.delta
# decreases the longer nodes are present in the network.

pdf("Soft Constraints.pdf",width=10,height=5)

adj=adjO=matrix(0,nrow=3,ncol=3)
adj[upper.tri(adj)]=1
Q=qgraph(adj,vsize=3,height=5,width=10,layout="spring",
esize=1,filetype='',directed=T)
cons=Q$layout
for (i in 1:20)
{
x=nrow(adj)
adjN=matrix(0,nrow=x+1,ncol=x+1)
adjN[1:x,1:x]=adj
consN=matrix(NA,nrow=x+1,ncol=2)
consN[1:x,]=cons[1:x,]
layout.par=list(init=rbind(cons,c(0,0)),
max.delta=10/(x+1):1,area=10^2,repulse.rad=10^3)
y=sample(c(x,sample(1:(x),1)),1)
adjN[y,x+1]=1
Q=qgraph(adjN,Q,layout="spring",layout.par=layout.par)
cons=Q$layout
adj=adjN
}
dev.off()

## End(Not run)
```

qgraph.loadings *qgraph.loadings*

Description

This function is a wrapper function for qgraph designed to visualize factor loadings.

Usage

```
qgraph.loadings( fact, ...)
```

Arguments

fact	A matrix containing factor loadings (items per row, factors per column) or an "loadings" object
...	Additional optional arguments passed to qgraph and special arguments used in this function (described below).

Additional optional arguments

layout If "default" a standard layout for factor models will be made. If this is "circle" the default layout is circled (factors in the centre, items at the edge). No other layouts are currently supported.

vsize A vector where the first value indicates the size of manifest variables and the second value indicates the size of latent variables.

model "reflective" to have arrows go to manifest variables, "formative" to have arrows go to latent variables or "none" (default) for no arrows

crossloadings Logical, if TRUE then for each manifest variable the strongest loading is omitted (default to FALSE).

groups An optional list containing the measurement model, see qgraph

Fname When there is only one factor, this is it's name. If there are more factors, the names in the groups list are used only if the factors can be identified.

resid Values for the residuals

residSize Size of the residuals, defaults to 0.1

factorCors Correlation matrix of the factors

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

See Also

qgraph qgraph.pca qgraph.efa

Examples

```
## Not run:
# Load big5 dataset:
data(big5)
data(big5groups)

big5efa <- factanal(big5,factors=5,rotation="promax",scores="regression")
big5loadings <- loadings(big5efa)
qgraph.loadings(big5loadings,groups=big5groups,rotation="promax",minimum=0.2,
cut=0.4,vsize=c(1.5,15),borders=FALSE,vTrans=200)

# Tree layout:
qgraph.loadings(big5loadings,groups=big5groups,rotation="promax",minimum=0.2,
cut=0.4,vsize=c(1.5,15),borders=FALSE,vTrans=200,
layout="tree",width=20,filetype="R")

## End(Not run)
```

---

qgraph.panel *qgraph.panel*

Description

Creates a 4-panel graph. See details. The usage is the same as qgraph

Usage

```
qgraph.panel(input, ...)
```

Arguments

input	The weights matrix or edgelist. See qgraph. Correlation matrix is recommended.
...	Optional additional arguments (only 'layout' and 'graph' are omitted). See qgraph.

Details

This function will create a 4-panel plot containing four plots useful in analyzing correlation matrices:

1. Association graph with circular layout
2. Association graph with spring layout
3. Concentration graph with spring layout
4. Factorial graph with spring layout

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

See Also

qgraph qgraph.svg

Examples

```
## Not run:
data(big5)
data(big5groups)

qgraph.panel(cor(big5),groups=big5groups,minimum=0.2,borders=FALSE,vsize=1,cut=0.3)

## End(Not run)
```

---

qgraph.pca *qgraph.pca*

Description

This function performs an Principal Component Analysis (PCA) using the 'princomp' function of the psych package (Revelle, 2010) and sends the acquired factor loadings to qgraph.loadings.

Usage

```
qgraph.pca( cor, factors=1, ..., rotation="promax", factorCors = TRUE)
```

Arguments

cor	A correlation matrix or a "principal" object
factors	The number of factors to extract
...	arguments passed to qgraph.loadings
rotation	rotation to be used. Can be "varimax", "promax" or "none"
factorCors	Logical, should the correlations between factors be plotted? Defaults to TRUE

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Revelle, W. (2010) psych: Procedures for Personality and Psychological Research Northwestern University, Evanston, http://personality-project.org/r/psych.manual.pdf, 1.0-93

See Also

qgraph qgraph.efa qgraph.loadings

Examples

```
## Not run:
data(big5)
data(big5groups)

qgraph.pca(cor(big5),5,groups=big5groups,rotation="promax",minimum=0.2,
cut=0.4,vsize=c(1,7),borders=FALSE,vTrans=200)

# Tree layout:
qgraph.pca(cor(big5), 5,groups=big5groups, rotation="promax", minimum=0.2,
  cut=0.4, vsize=c(1.5,15), borders=FALSE, layout="tree", width=20, filetype="R")

## End(Not run)
```

---

qgraph.sem *qgraph: Structural Equation Modelling*

Description

This functions uses a "sem" object from the sem function (from the sem package; Fox, 2010) and outputs a multi-page pdf file containing goodness of fit indices, path diagram, graphs of the parameter estimates and graphs of the implied and observed covariance and correlation matrices.

Usage

```
qgraph.sem(res, layout="circle",  ..., vsize.man=3, vsize.lat=6, filename="qgraph",
filetype="pdf", residuals=TRUE, panels=2, include=1:12, latres=TRUE,
curve=0, residSize=0.2, onefile=TRUE, width = 7, height = 7)
```

Arguments

res	The output from the 'sem' function. See details for extra details on specifying the model
layout	The layout used for the path diagram. Can be "tree", "spring", "circle" and "springtree"

`...`	arguments passed to qgraph. This is both for the path diagram and for the correlation/covariance plots.
`vsize.man`	Size of the manifest variables in the path diagram
`vsize.lat`	Size of the latent variables in the path diagram
`filename`	Name of the file (will be extended with the filetype)
`filetype`	The filetype to be used. Can be "pdf" to make a pdf (default) or anything else to plot in R. More filetypes will be supported in a future version.
`residuals`	Logical indicating if the residuals should be included in the path diagram. If this is FALSE then residual variances will be shown as colors on the nodes. Default is TRUE
`panels`	The number of panels to be used in the pdf file. Can be 1, 2, 4 or 8
`include`	A vector indicating which panels should be included in the output
`latres`	This is currently not supported, leave to TRUE
`curve`	Numerical value indicating the curve of edges that are on the same level in the "tree" layout, See details. This represent an offset from the middle of the straight edge through where the curved edge must be drawn. 0 indicates no curve, and any other value indicates a curve of that strength. Defaults to 0.2
`residSize`	Size of the residual arrows
`onefile`	Logical indicating if a multi-page pdf should be produced. If FALSE each plot will be a new pdf. Use this only with panels=1 and filename="(Arbitrary name)%03d".
`width`	Width of each panel, in inches
`height`	Height of each panel, in inches

Details

This function uses a "sem" object and outputs a multi-page pdf file. The function reads the 'sem' file and creates a residual variable for each variable present in the model. The default layout is a tree-layout (layout="tree") in which each variable is placed as a node on one of four vertical levels. At the bottom are the residuals of the manifest variables placed, Above that the manifest variables, above that the latent variables and at the top the residuals of the latent variables. The nodes are evenly spaced horizontally in order of appearance in the model (residuals are placed at the same horizontal position of their corresponding variable). So the order of specifying in the model defines the order of placement in the path diagram. If the 'residuals' argument is FALSE then residuals are omitted and this diagram will only contain two levels.

Alternatively the 'spring' layout can be used (layout="spring"). This will use the Fruchterman-reingold algorithm (Fruchterman & Reingold, 1991) is used, which has been ported from the 'sna' package (Butts, 2010). This is a force-directed algorithm. It is best to use this in combination with residuals=FALSE.

Names for variables and edges used in the model specification are passed to the path diagram. To keep the model readable it is advised to limit these names to 3 characters. If a variable or edge name contains an asterisk, the asterisk is omitted and the name will be printed in the symbol font. This way Greek letters can be used (e.g. the edge name "l*" will make a lambda character). Because the symbol font conveniently uses Arabic numerals, parameter names like beta1 can be created with "*b1" in the model.

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Carter T. Butts <buttsc@uci.edu> (2010). sna: Tools for Social Network Analysis. R package version 2.2-0. http://CRAN.R-project.org/package=sna

John Fox with contributions from Adam Kramer <jfox@mcmaster.ca> and Michael Friendly (2010). sem: Structural Equation Models. R package version 0.9-21. http://CRAN.R-project.org/package=sem

Fruchterman, T. & Reingold, E. (1991). Graph drawing by force-directed placement. Software - Pract. Exp. 21, 1129?1164.

See Also

qgraph qgraph.cfa qgraph.semModel sem

Examples

```
## Not run:

require('sem')

# This example is taken from the examples of the sem function.
# Only names were changed to better suit the path diagram.

# ---------------------- Thurstone data ---------------------------------------
#  Second-order confirmatory factor analysis, from the SAS manual for PROC CALIS

R.thur <- readMoments(diag=FALSE, names=c('Sen','Voc',
        'SC','FL','4LW','Suf',
        'LS','Ped', 'LG'))
    .828
    .776   .779
    .439   .493    .46
    .432   .464    .425   .674
    .447   .489    .443   .59    .541
    .447   .432    .401   .381    .402   .288
    .541   .537    .534   .35    .367   .32   .555
    .38   .358    .359   .424    .446   .325   .598   .452

model.thur <- specifyModel()
    F1 -> Sen,               *l11, NA
    F1 -> Voc,               *l21, NA
    F1 -> SC,                *l31, NA
    F2 -> FL,                *l41, NA
    F2 -> 4LW,               *l52, NA
    F2 -> Suf,               *l62, NA
```

```
    F3 -> LS,               *l73, NA
    F3 -> Ped,              *l83, NA
    F3 -> LG,               *l93, NA
    F4 -> F1,               *g1,  NA
    F4 -> F2,               *g2,  NA
    F4 -> F3,               *g3,  NA
    Sen <-> Sen,            q*1,   NA
    Voc<-> Voc,             q*2,   NA
    SC <-> SC,              q*3,   NA
    FL <-> FL,              q*4,   NA
    4LW <-> 4LW,            q*5,   NA
    Suf<-> Suf,             q*6,   NA
    LS <-> LS,              q*7,   NA
    Ped<-> Ped,             q*8,   NA
    LG <-> LG,              q*9,   NA
    F1 <-> F1,              NA,     1
    F2 <-> F2,              NA,     1
    F3 <-> F3,              NA,     1
    F4 <-> F4,              NA,     1

sem.thur <- sem(model.thur, R.thur, 213)

# Run qgraph:
qgraph.sem(sem.thur,filename="Thurstone tree",layout="tree",edge.label.cex=0.6,
  curve=0.4,width=8,height=5)

# Spring layout:
qgraph.sem(sem.thur,filename="Thurstone spring",layout="spring",residuals=FALSE,
  width=5,height=5)
## End(Not run)
```

`qgraph.semModel` *qgraph: SEM model pathdiagram*

Description

This is a watered down version of `qgraph.sem` that only plots a diagram. It is called if a "mod" object is supplied to `qgraph`

Usage

```
qgraph.semModel(mod, manifest = NULL, layout = "spring", vsize.man = 3,
  vsize.lat = 6, residuals = TRUE, latres = TRUE, curve = 0.2, residSize = 0.2,
  ...)
```

Arguments

`mod`	A "mod" object (model of the sem package (Fox; 2010) or a "sem" object

manifest	Vector containing the names of the manifest variables
layout	The layout used for the path diagram. Can be "tree", "spring", "circle" and "springtree". Defaults to "spring"
vsize.man	Size of the manifest variables in the path diagram
vsize.lat	Size of the latent variables in the path diagram
residuals	Logical indicating if the residuals should be included in the path diagram. If this is FALSE then residual variances will be shown as colors on the nodes. Default is TRUE
latres	This is currently not supported, leave to TRUE
curve	Numerical value indicating the curve of edges that are on the same level in the "tree" layout, See details. This represent an offset from the middle of the straight edge through where the curved edge must be drawn. 0 indicates no curve, and any other value indicates a curve of that strength. Defaults to 0.2
residSize	Size of the residual arrows
...	Arguments passed to qgraph

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

John Fox with contributions from Adam Kramer <jfox@mcmaster.ca> and Michael Friendly (2010). sem: Structural Equation Models. R package version 0.9-21. http://CRAN.R-project.org/package=sem

See Also

qgraph

Examples

```
## Not run:

require('sem')

# This example is taken from the examples of the sem function.
# Only names were changed to better suit the path diagram.

# ---------------------- Thurstone data ---------------------------------------
#  Second-order confirmatory factor analysis, from the SAS manual for PROC CALIS

R.thur <- readMoments(diag=FALSE, names=c('Sen','Voc',
        'SC','FL','4LW','Suf',
        'LS','Ped', 'LG'))
    .828
```

```
    .776   .779
    .439   .493    .46
    .432   .464    .425   .674
    .447   .489    .443   .59    .541
    .447   .432    .401   .381    .402   .288
    .541   .537    .534   .35    .367   .32   .555
    .38   .358    .359   .424    .446   .325   .598   .452

model.thur <- specifyModel()
    F1 -> Sen,               *l11, NA
    F1 -> Voc,               *l21, NA
    F1 -> SC,                *l31, NA
    F2 -> FL,                *l41, NA
    F2 -> 4LW,               *l52, NA
    F2 -> Suf,               *l62, NA
    F3 -> LS,                *l73, NA
    F3 -> Ped,               *l83, NA
    F3 -> LG,                *l93, NA
    F4 -> F1,                *g1,  NA
    F4 -> F2,                *g2,  NA
    F4 -> F3,                *g3,  NA
    Sen <-> Sen,             q*1,   NA
    Voc<-> Voc,              q*2,   NA
    SC <-> SC,               q*3,   NA
    FL <-> FL,               q*4,   NA
    4LW <-> 4LW,             q*5,   NA
    Suf<-> Suf,              q*6,   NA
    LS <-> LS,               q*7,   NA
    Ped<-> Ped,              q*8,   NA
    LG <-> LG,               q*9,   NA
    F1 <-> F1,               NA,     1
    F2 <-> F2,               NA,     1
    F3 <-> F3,               NA,     1
    F4 <-> F4,               NA,     1



# Run qgraph:
qgraph(model.thur)

# Tree layout:
qgraph(model.thur,layout="tree",manifest=c('Sen','Voc','SC','FL','4LW','Suf','LS','Ped', 'LG'))


## End(Not run)
```

---

qgraph.svg *qgraph.svg*

Description

IMPORTANT NOTE: RSVGTipsDevice is a 32-bit only package, so SVG functionality is not available in 64bit versions of R.

This function makes a series of SVG images with hyperlinks to each other using the RSVGTipsDevice package (Plate, 2009). The arguments are the same as qgraph except that some arguments can be assigned a vector with multiple options.

Usage

```
qgraph.svg( input, layout=c( "circular", "spring" ),
graph=c( "association", "concentration", "factorial" ),
cut=c( 0, 0.2, 0.3, 0.5), filename="qgraph", title="qgraph output",
nfact=round(ncol(input)/2,0),tooltips=NULL,... )
```

Arguments

input	A weights matrix (correlation matrix is recommended). See qgraph
layout	The layout of the graph, see qgraph, unlike qgraph it is not possible to assign a specific layout.
graph	The graph to be made based on the correlation matrix. See qgraph
cut	The cutoff score. See qgraph
filename	The name of the SVG files.
title	The title that will be printed at the top of the pictures
nfact	The number of factors that will be used in the EFA that makes the factorial graph. See qgraph
tooltips	A vector with for each node the tooltip to be used.
...	Additional arguments that are passed to qgraph

Details

This function works the same as qgraph except that the arguments 'layout', 'graph' and 'cut' can be given a vector with multiple options. The RSVGTipsDevice package (Plate, 2009) is used to accomplish this. A SVG picture will be created for each combination of the arguments 'layout', 'graph' and 'cut', with hyperlinks between them

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

References

Sacha Epskamp, Angelique O. J. Cramer, Lourens J. Waldorp, Verena D. Schmittmann, Denny Borsboom (2012). qgraph: Network Visualizations of Relationships in Psychometric Data. Journal of Statistical Software, 48(4), 1-18. URL http://www.jstatsoft.org/v48/i04/.

Plate, T. <tplate@acm.org> and based on RSvgDevice by T Jake Luciani <jakeluciani@yahoo.com> (2009). RSVGTipsDevice: An R SVG graphics device with dynamic tips and hyperlinks. R package version 1.0-1.

See Also

qgraph

Examples

```
## Not run:
#### VISUALIZE CORRELATION MATRIX ###
eta=matrix(rnorm(200*5),ncol=5)
lam=matrix(0,nrow=100,ncol=5)
for (i in 1:5) lam[(20*i-19):(20*i),i]=rnorm(20,0.7,0.3)
eps=matrix(rnorm(200*100),ncol=100)
Y=eta%*%t(lam)+eps

tooltips=paste("item",1:100)
groups=list(1:20,21:40,41:60,61:80,81:100)
names(groups)=paste("Factor",LETTERS[1:5])
# Run qgraph:
qgraph.svg(cor(Y),groups=groups,tooltips=tooltips,vsize=3)
## End(Not run)
```

---

summary.qgraph *Summary method for "qgraph"*

Description

This function creates a brief summary based on a "qgraph" object.

Usage

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

Arguments

object	A "qgraph" object
...	These arguments are not used

Author(s)

Sacha Epskamp (qgraph@sachaepskamp.com)

See Also

qgraph

Index