Package: concorR (via r-universe)

September 13, 2024

Type Package

Title CONCOR and Supplemental Functions

Version 0.2.1.9000

Description Contains the CONCOR (CONvergence of iterated CORrelations) algorithm and a series of supplemental functions for easy running, plotting, and blockmodeling. The CONCOR algorithm is used on social network data to identify network positions based off a definition of structural equivalence; see Breiger, Boorman, and Arabie (1975) <doi:10.1016/0022-2496(75)90028-0> and Wasserman and Faust's book Social Network Analysis: Methods and Applications (1994). This version allows multiple relationships for the same set of nodes and uses both incoming and outgoing ties to find positions.

Imports igraph, sna, stats, graphics, viridis

License GPL (>=2)

Encoding UTF-8

LazyData true

URL `https://github.com/ATraxLab/concorR`

BugReports `https://github.com/ATraxLab/concorR/issues`

RoxygenNote 7.1.1

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

RemoteUrl https://github.com/atraxlab/concorr

RemoteRef HEAD

RemoteSha 1ca6eb1daf399a135b56d9eece5da9a21c563e3d

Contents

concor *Find CONCOR partition for a graph*

Description

Use the CONCOR (CONvergence of iterated CORrelations) algorithm to identify roles within social network data.

Usage

```
concor(m_list, nsplit = 1, self_ties = FALSE, cutoff = .9999999, max_iter = 50)
```

Arguments

`m_list`	A list of adjacency matrices. Matrices must be square, of the same dimensions, and have row/column names (node names). Each matrix represents a different relationship of the same nodes. If there is only one relationship of interest, `m_list` is a list of that single matrix.
`nsplit`	The number of times the input matrices will be partitioned. Each split divides a partition in two, resulting in 2^`nsplit` roles identified.
`self_ties`	A Boolean representing whether self-ties can be present in the data. `TRUE` allows for self-ties; `FALSE` does not.
`cutoff`	The cutoff point of the iterated correlations. Both this and `max_iter` can be lowered slightly to improve speed.
`max_iter`	The maximum number of times the correlation will be run for a split. Both this and `cutoff` can be lowered slightly to improve speed.

Details

This version works for multiple relationships, assuming they all are for the same data (same size of input matrices), and can be used with isolates present. It requires further testing on weighted networks (but appears to successfully split such networks). It will attempt to split the network `nsplit` times, causing there to be 2^`nsplit` partitions, plus one for isolated nodes (if they exist), unless a structurally equivalent node group or singular node group is present. If the algorithm attempts to split such a node group the function will ignore that group and continue to split all other blocks until specified. If a higher number of splits, `nsplit`, are requested than are possible to apply to the specified data (due to structurally equivalent node groups being present or all blocks being singular nodes) the code will warn the user that split `nsplit` was the same as split `i`, the final possible split, and stop.

Value

A data frame with two columns: `block` is the block or role identified by CONCOR, and `vertex` is the node names.

References

R. L. Breiger, S. A. Boorman, P. Arabie, An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *J. of Mathematical Psychology*. **12**, 328 (1975). doi:10.1016/00222496(75)900280

S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* (Cambridge University Press, 1994).

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(a) <- c("a", "b", "c", "d")
colnames(a) <- c("a", "b", "c", "d")

concor(list(a))
```

`concor1` *Find +1/-1 convergence points of an input matrix*

Description

A subfunction of `concor`.

Runs the Pearson correlation on the matrix stack repeatedly until a square matrix consisting of only +1 and -1 is formed.

Usage

```
concor1(m_stack, cutoff = .9999999, max_iter = 50)
```

Arguments

`m_stack` The stack of input matrices, which must include all transposes and relations.

`cutoff` The absolute value of the convergence threshold for each matrix element.

`max_iter` Maximum number of times to run the correlation while seeking convergence.

Details

For network data with R relations, the input matrix "stack" is a (2 x N x R) x N matrix, consisting of each relation's adjacency matrix, then the transpose of that matrix, appended to each other vertically. The correlation is run until either all matrix entries have absolute values greater than cutoff or the maximum number of iterations max_iter is reached. On its own, this function does not execute the whole CONCOR method, but is listed separately from concor to demonstrate the core step.

Value

A square matrix, with number of rows/columns equal to the number of columns of m_stack, where the values are either +1 or -1. The input matrix can later be split into two "positions" based off the locations of the positive/negative values.

See Also

concor

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
b <- rbind(a, t(a))

concor1(b)
```

concorR *The concorR Package*

Description

concorR implements the CONCOR (CONvergence of iterated CORrelations) algorithm as introduced by Breiger, Boorman, and Arabie (1975) and detailed by Wasserman and Faust (1994). This package includes the concor algorithm itself, and some useful functions for plotting and interpreting the outputs. The plotting methods included are: plotting the overall network with the concor partitions as colors (using the **igraph** package), plotting a blockmodel using the concor partitions (using the **sna** package), and plotting reduced network graphs showing the connections between concor partitions (again using **igraph**).

Author(s)

Tyme Suda <suda.4@wright.edu>

Adrienne Traxler <adrienne.traxler@wright.edu>

This work was supported by National Science Foundation awards DUE-1712341 and DUE-1711017.

References

R. L. Breiger, S. A. Boorman, P. Arabie, An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *J. of Mathematical Psychology*. **12**, 328 (1975). doi:10.1016/00222496(75)900280

S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* (Cambridge University Press, 1994).

concor_igraph_apply *Find CONCOR partition and add to a list of* **igraph** *objects*

Description

Takes a list of **igraph** objects, runs concor until the desired number of splits, and adds the final split to each object as a vertex attribute.

Usage

```
concor_igraph_apply(igraph_list, nsplit = 1)
```

Arguments

igraph_list	The list of **igraph** objects for use in concor.
nsplit	Number of times to split each network.

Details

This function is a wrapper for a common task: Running concor on one or more **igraph** objects and adding the resulting partition to each object as a vertex attribute. If multiple **igraph** objects are included in the input list, they should be multiple relations for the same nodes.

If all of the input graphs are weighted, edge weights will be used by concor.

Value

Returns a list of **igraph** objects, each with a vertex attribute csplitn (where n is nsplit) that contains the block assignment obtained from concor.

See Also

concor, concor_make_igraph

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(a) <- c("a", "b", "c", "d")
colnames(a) <- c("a", "b", "c", "d")
a.igraph <- igraph::graph_from_adjacency_matrix(a)
concor_igraph_apply(list(a.igraph))

b <- matrix(c(0, 0, 0, 4, 1, 0, 3, 0, 2, 1, 0, 1, 1, 0, 2, 0), ncol = 4)
rownames(b) <- c("a", "b", "c", "d")
colnames(b) <- c("a", "b", "c", "d")
b.igraph <- igraph::graph_from_adjacency_matrix(b, weighted = "weight")
concor_igraph_apply(list(b.igraph))

concor_igraph_apply(list(a.igraph, b.igraph))
```

---

concor_make_igraph *Find CONCOR partition and create a list of* **igraph** *objects*

Description

Run concor on a list of adjacency matrices and create a list of **igraph** objects with a specific CONCOR split added as a vertex attribute.

Usage

```
concor_make_igraph(adj_list, nsplit = 1)
```

Arguments

adj_list	A list of adjacency matrices, each representing a different relation for the same nodes.
nsplit	The total number of splits CONCOR will (attempt to) perform.

Details

This function is a wrapper for a common task: Running concor on one or more adjacency matrices, then saving the networks as **igraph** objects with the CONCOR partition as a vertex attribute. If multiple adjacency matrices are included in the input list, they should be multiple relations for the same nodes.

If any of the input matrices have entries other than 0 and 1, all are treated as weighted (a weight edge attribute is added to each output graph).

Value

Returns a list of **igraph** objects with a vertex attribute csplitn, where n is nsplit that contains the block assignment obtained from concor.

See Also

concor, concor_igraph_apply

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(a) <- c("a", "b", "c", "d")
colnames(a) <- c("a", "b", "c", "d")
concor_make_igraph(list(a))

b <- matrix(c(0, 0, 0, 4, 1, 0, 3, 0, 2, 1, 0, 1, 1, 0, 2, 0), ncol = 4)
rownames(b) <- c("a", "b", "c", "d")
colnames(b) <- c("a", "b", "c", "d")
concor_make_igraph(list(b))

concor_make_igraph(list(a, b))
```

krack_advice *Krackhardt High-Tech Managers data*

Description

A network of connections between 21 managers at a manufacturing company taken by David Krackhardt in 1987. The three networks available are advice, friendship, and report structure.

Usage

```
krack_advice

krack_friend

krack_report
```

Format

A directed and unweighted **igraph** object.

Vertex attributes:

- Name: Vertex names (character; "v1", "v2", ..., "v21").
- x,y: A set of plotting coordinates, used if no other layout is supplied (numeric).
- Age: The age of each manager (integer).
- Tenure: The tenure of each manager (numeric).
- Level: The level in the corporate hierarchy (integer; 1 = CEO, 2 = Vice President, 3 = manager).
- Department: What department each manager is in (integer; 1, 2, 3, 4, or 0 for the CEO).

Details

Edges in the krack_advice and krack_friend networks come survey questions answered by all 21 managers. The krack_report network uses the formal organization chart to define connections between managers.

Source

The data was found at http://vlado.fmf.uni-lj.si/pub/networks/data/WaFa/default.htm and reformatted for use in R.

References

D. Krackhardt, Cognitive social structures. *Social Networks*. **9**, 104 (1987). doi:10.1016/0378-8733(87)900098

S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* (Cambridge University Press, 1994).

make_blk *Make* blockmodel *objects using CONCOR partition*

Description

Use the CONCOR algorithm to find network positions, continuing through a specified number of splits, and output blockmodel object(s).

Usage

```
make_blk(adj_list, nsplit = 1)
```

Arguments

adj_list	A list of adjacency matrices, each representing a different relation for the same nodes.
nsplit	The total number of splits CONCOR will (attempt to) perform.

Details

Runs concor to a specified number of splits (nsplit) and creates a blockmodel type object from each input matrix in adj_list using the membership vector generated by concor. This object holds summary information of interest, such as the reduced density matrix, as well as the permuted adjacency matrix that shows the blocked structure.

See the help file for blockmodel in the **sna** package for more details.

Value

A list of blockmodel type objects, one for each input adjacency matrix. Each blockmodel in the list corresponds to one relation.

See Also

blockmodel, concor

Examples

```
g1 <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(g1) <- c("a", "b", "c", "d")
colnames(g1) <- c("a", "b", "c", "d")

g2 <- matrix(c(0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0), ncol = 4)
rownames(g2) <- c("a", "b", "c", "d")
colnames(g2) <- c("a", "b", "c", "d")

gl <- list(g1, g2)
make_blk(gl)
```

---

make_reduced — *Run CONCOR and output reduced adjacency matrices*

Description

Takes a list of adjacency matrices, partitions using concor, and returns a list of reduced adjacency matrices and their corresponding cutoff densities. Each reduced matrix corresponds to one input relation. The **sna** package must be installed, but does not need to be attached.

Usage

```
make_reduced(adj_list, nsplit = 1, stat = 'density')
```

Arguments

adj_list	A list of adjacency matrices, each representing a different relation for the same nodes.
nsplit	The total number of splits CONCOR will (attempt to) perform.
stat	The statistic that will be used to determine if edges should be included in a reduced network. Default value is density

Details

A reduced network represents each identified position in the network as a single node. Links (or self-links) exist if the density (or normalized degree) of ties from that block to a target block is greater than a threshold density (or degree). In the default implementation, the density of the whole network is used as the threshold for each block. In the degree implementation, the normalized degree of the network is used as the threshhold.

In the list of input matrices adj_list, each should correspond to a different relation for the same nodes. Each adjacency matrix is partitioned with the CONCOR algorithm, continuing for nsplit divisions. After the threshold density is applied, each entry in the reduced matrix has values of 0 or 1.

Value

`reduced_mat` A list of reduced matrices, one for each input matrix.

`dens` A vector of the cut-off densities used (equal to the edge density of each entry in `adj_list`). Only for `stat="density"`

`deg` A vector of the cut-off normalized degrees used (equal to the mean normalized degree of each entry in `adj_list`). Only for `stat="degree"`.

References

S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* (Cambridge University Press, 1994).

See Also

`concor`, `make_blk`

Examples

```
g1 <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(g1) <- c("a", "b", "c", "d")
colnames(g1) <- c("a", "b", "c", "d")

g2 <- matrix(c(0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0), ncol = 4)
rownames(g2) <- c("a", "b", "c", "d")
colnames(g2) <- c("a", "b", "c", "d")

make_reduced(list(g1, g2), nsplit = 1)
make_reduced(list(g1, g2), nsplit = 1, stat="degree")
```

`make_reduced_from_partition`

Output reduced adjacency matrices given a partitioning of the nodes into appropriate groups

Description

Takes an adjacency matrix and a partition of the nodes, and returns a reduced adjacency matrix and the corresponding cutoff density/degree. The **sna** package must be installed, but does not need to be attached.

Usage

```
make_reduced_from_partition(adj_mat, partition, stat = 'density')
```

Arguments

`adj_mat` A adjacency matrix

`partition` A vector that gives the desired partitioning of the nodes. This must be an integer vector with all values between 1 and `max(partition)`.

`stat` The statistic that will be used to determine if edges should be included in a reduced network. Default value is `density`. Possible values are `density` and `degree`.

Details

A reduced network represents each identified position in the network as a single node. Links (or self-links) exist if the density (or normalized degree) of ties from that block to a target block is greater than a threshold density (or degree). In the default implementation, the density of the whole network is used as the threshold for each block. In the degree implementation, the normalized degree of the network is used as the threshhold.

Value

`reduced_mat` A reduced matrix

`dens` The cut-off density used (equal to the edge density of `adj_mat`). Only for `stat="density"`

`deg` The cut-off normalized outdegree used (equal to the mean outdegree of `adj_mat`). Only for `stat="degree"`.

References

S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* (Cambridge University Press, 1994).

See Also

`make_reduced`, `make_blk`

Examples

```
## Generate a random network (with reproducibility)
set.seed(1234)
g  <-  igraph::erdos.renyi.game(50,p=0.2)
g_adj  <- as.matrix(igraph::as_adjacency_matrix(g))
ebc.g  <- igraph::edge.betweenness.community(g)
ebPart  <- ebc.g$membership

## Generate reduced network using degree statistic
g.red  <- make_reduced_from_partition(g_adj, ebPart, stat='degree')
plot_reduced(make_reduced_igraph(g.red$reduced_mat))

## Generate reduced network using density statistic
g.red.den  <- make_reduced_from_partition(g_adj, ebPart, stat='density')
plot_reduced(make_reduced_igraph(g.red.den$reduced_mat))
```

make_reduced_igraph *Build an igraph object for a reduced network*

Description

Turns a reduced adjacency matrix (usually output from make_reduced) into an igraph object. This function requires **igraph** to work.

Usage

```
make_reduced_igraph(reduced_mat)
```

Arguments

reduced_mat	A reduced network adjacency matrix (typically outputted from make_reduced in the reduced_mat list)

Value

A directed and unweighted igraph object for the reduced matrix.

See Also

make_reduced

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(a) <- c("a", "b", "c", "d")
colnames(a) <- c("a", "b", "c", "d")

r_mat <- make_reduced(list(a), nsplit = 1)
make_reduced_igraph(r_mat$reduced_mat[[1]])
```

plot_blk *Plot a blockmodel*

Description

Displays a plot of a blockmodel. Based on plot.blockmodel (**sna** must be installed), but reformats the plot to be square and removes the mandatory title.

Usage

```
plot_blk(x, labels = FALSE, ...)
```

Arguments

`x` An object of class `blockmodel`

`labels` If `TRUE`, vertex ids are displayed as row/column/diagonal labels. If `FALSE`, no node labels are shown.

`...` Further arguments passed to or from other methods

Details

This is a modification of the `plot.blockmodel` function. The original displays vertex ids as row, column, and diagonal labels, which can be unreadable for larger networks. `plot.blockmodel` also adds a title of the form "Relation - 1", which this version omits.

Value

Returns `NULL`, invisibly.

Author(s)

Carter T. Butts (buttsc@uci.edu)

Modified by Tyme Suda

References

Carter T. Butts (2019). sna: Tools for Social Network Analysis. R package version 2.5. `https://CRAN.R-project.org/package=sna`

See Also

`plot.blockmodel`

Examples

```
g1 <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(g1) <- c("a", "b", "c", "d")
colnames(g1) <- c("a", "b", "c", "d")

gl <- list(g1)
bm <- make_blk(gl, 1)[[1]]
plot_blk(bm)
```

plot_reduced *Plot a reduced network*

Description

Plot a reduced network using **igraph**, with nodes colored by block number.

Usage

```
plot_reduced(iobject,main='')
```

Arguments

`iobject` An `igraph` object representing the reduced network.

`main` Text for the title of the plot. Defaults to an empty string. See `plot`.

Details

Plots the reduced network, where each node represents all the nodes assigned to that position by CONCOR. Node colors on the reduced network plot are assigned by position, so if CONCOR is also used for vertex color on a sociogram (as in `plot_socio`), the node colors will align between the plots.

Value

Returns `NULL`, invisibly.

See Also

`make_reduced`, `make_reduced_igraph`, `plot_socio`

Examples

```
library(igraph)
g1 <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(g1) <- c("a", "b", "c", "d")
colnames(g1) <- c("a", "b", "c", "d")

r_mat <- make_reduced(list(g1), nsplit = 1)
r_igraph <- make_reduced_igraph(r_mat$reduced_mat[[1]])

plot_reduced(r_igraph)
```

plot_socio *Plot sociogram colored by CONCOR partition*

Description

Plots a network of interest using CONCOR partition as vertex color. Uses an `igraph` object, normally created by `concor_make_igraph` or `concor_igraph_apply`.

Usage

```
plot_socio(iobject, nsplit = NULL)
```

Arguments

`iobject`	An `igraph` object with `concor` split as appropriately-named vertex attribute.
`nsplit`	Split number to use as vertex color.

Details

This is a shortcut to plot an `igraph` object with usually-readable settings. It looks for the input `iobject` to have a vertex attribute called `csplit(nsplit)` that holds the CONCOR partition assignment (for example, if `nsplit` = 2, then `plot_socio` expects a vertex attribute named `csplit2`).

Value

Returns NULL, invisibly.

See Also

`concor`, `concor_make_igraph`, `concor_igraph_apply`

Examples

```
a <- matrix(c(0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0), ncol = 4)
rownames(a) <- c("a", "b", "c", "d")
colnames(a) <- c("a", "b", "c", "d")

i_out <- concor_make_igraph(list(a))
plot_socio(i_out[[1]], 1)
```

Index