Package ‘assortnet’

July 22, 2025

Type Package

Title Calculate the Assortativity Coefficient of Weighted and Binary Networks

Version 0.20

Date 2023-02-24

Author Damien Farine `<damien.farine@anu.edu.au>`

Maintainer Damien Farine `<damien.farine@anu.edu.au>`

Description Functions to calculate the assortment of vertices in social networks. This can be measured on both weighted and binary networks, with discrete or continuous vertex values.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2023-02-24 12:30:06 UTC

Contents

`assortnet-package` *Calculate the assortativity coefficient of weighted and binary networks ~~ assortnet ~~*

Description

Functions to calculate the assortment of vertices in social networks. This can be measured on both weighted and binary networks, with discrete or continuous vertex values.

Details

Package:	assortnet
Type:	Package
Version:	0.20
Date:	2023-02-24
License:	GPL2

Author(s)

Maintainer: Damien Farine <damien.farine@anu.edu.au>

References

Newman (2003) Mixing patterns in networks. Physical Review E (67) Farine, D.R. (2014) Measuring phenotypic assortment in animal social networks: weighted associations are more robust than binary edges. Animal Behaviour 89: 141-153.

---

assortment.continuous *Assortment on continuous vertex values*

Description

Calculates the assortativity coefficient for weighted and unweighted graphs with numerical vertex values

Usage

```
assortment.continuous(graph, vertex_values, weighted = TRUE,
SE = FALSE, M = 1, na.rm = FALSE)
```

Arguments

`graph`	A Adjacency matrix, as an N x N matrix. Can be weighted or binary.
`vertex_values`	Values on which to calculate assortment, vector of N numbers
`weighted`	Flag: TRUE to use weighted edges, FALSE to turn edges into binary (even if weights are given)
`SE`	Calculate standard error using the Jackknife method.
`M`	Binning value for Jackknife, where M edges are removed rather than single edges. This helps speed up the estimate for large networks with many edges.
`na.rm`	Remove all nodes which have NA as vertex_values from both the network and the vertex_values object. If this is False and NAs are present, an error message will be displayed.

Value

This function returns a named list, with two elements:

$r the assortativity coefficient $SE the standard error

Author(s)

Damien Farine dfarine@orn.mpg.de

References

Newman (2003) Mixing patterns in networks. Physical Review E (67) Farine, D.R. (2014) Measuring phenotypic assortment in animal social networks: weighted associations are more robust than binary edges. Animal Behaviour 89: 141-153.

Examples

```
# DIRECTED NETWORK EXAMPLE
# Create a random directed network
N <- 20
dyads <- expand.grid(ID1=1:20,ID2=1:20)
dyads <- dyads[which(dyads$ID1 != dyads$ID2),]
weights <- rbeta(nrow(dyads),1,15)
network <- matrix(0, nrow=N, ncol=N)
network[cbind(dyads$ID1,dyads$ID2)] <- weights

# Create random continues trait values
traits <- rnorm(N)

# Test for assortment as binary network
assortment.continuous(network,traits,weighted=FALSE)

# Test for assortment as weighted network
assortment.continuous(network,traits,weighted=TRUE)



# UNDIRECTED NETWORK EXAMPLE
# Create a random undirected network
N <- 20
dyads <- expand.grid(ID1=1:20,ID2=1:20)
dyads <- dyads[which(dyads$ID1 < dyads$ID2),]
weights <- rbeta(nrow(dyads),1,15)
network <- matrix(0, nrow=N, ncol=N)
network[cbind(dyads$ID1,dyads$ID2)] <- weights
network[cbind(dyads$ID2,dyads$ID1)] <- weights

# Create random continues trait values
traits <- rnorm(N)

# Test for assortment as binary network
assortment.continuous(network,traits,weighted=FALSE)
```

```
# Test for assortment as weighted network
assortment.continuous(network,traits,weighted=TRUE)
```

---

assortment.discrete *Assortment on discrete vertex values*

Description

Calculates the assortativity coefficient for weighted and unweighted graphs with nominal/categorical vertex values

Usage

```
assortment.discrete(graph, types, weighted = TRUE, SE = FALSE, M = 1, na.rm = FALSE)
```

Arguments

graph	Adjacency matrix, as an N x N matrix. Can be weighted or binary.
types	Values on which to calculate assortment, vector of N labels
weighted	Flag: TRUE to use weighted edges, FALSE to turn edges into binary (even if weights are given)
SE	Calculate standard error using the Jackknife method.
M	Binning value for Jackknife, where M edges are removed rather than single edges. This helps speed up the estimate for large networks with many edges.
na.rm	Remove all nodes which have NA as type from both the network and the types object. If this is False and NAs are present, an error message will be displayed.

Value

This function returns a named list, with three elements:

$r the assortativity coefficient $SE the standard error $mixing_matrix the mixing matrix with the distribution of edges or edge weights by category

Author(s)

Damien Farine dfarine@orn.mpg.de

References

Newman (2003) Mixing patterns in networks. Physical Review E (67) Farine, D.R. (2014) Measuring phenotypic assortment in animal social networks: weighted associations are more robust than binary edges. Animal Behaviour 89: 141-153.

Examples

```
# DIRECTED NETWORK EXAMPLE
# Create a random directed network
N <- 20
dyads <- expand.grid(ID1=1:20,ID2=1:20)
dyads <- dyads[which(dyads$ID1 != dyads$ID2),]
weights <- rbeta(nrow(dyads),1,15)
network <- matrix(0, nrow=N, ncol=N)
network[cbind(dyads$ID1,dyads$ID2)] <- weights

# Create random discrete trait values
traits <- rpois(N,2)

# Test for assortment as binary network
assortment.discrete(network,traits,weighted=FALSE)

# Test for assortment as weighted network
assortment.discrete(network,traits,weighted=TRUE)



# UNDIRECTED NETWORK EXAMPLE
# Create a random undirected network
N <- 20
dyads <- expand.grid(ID1=1:20,ID2=1:20)
dyads <- dyads[which(dyads$ID1 < dyads$ID2),]
weights <- rbeta(nrow(dyads),1,15)
network <- matrix(0, nrow=N, ncol=N)
network[cbind(dyads$ID1,dyads$ID2)] <- weights
network[cbind(dyads$ID2,dyads$ID1)] <- weights

# Create random discrete trait values
traits <- rpois(N,2)

# Test for assortment as binary network
assortment.discrete(network,traits,weighted=FALSE)

# Test for assortment as weighted network
assortment.discrete(network,traits,weighted=TRUE)
```

Index