Package ‘econetwork’

Type Package
Title Analyzing Ecological Networks
Version 0.5.1
Date 2021-02-26
Author Stephane Dray [ctb], Catherine Matias [ctb], Vincent Miele [aut, cre], Marc Ohlmann [aut], Giovanni Poggiato [ctb], Wilfried Thuiller [ctb]
Maintainer Vincent Miele <vincent.miele@univ-lyon1.fr>
Description A collection of advanced tools, methods and models specifically designed for analyzing different types of ecological networks - especially antagonistic (food webs, host-parasite), mutualistic (plant-pollinator, plant-fungus, etc) and competitive networks, as well as their variability in time and space. Statistical models are developed to describe and understand the mechanisms that determine species interactions, and to decipher the organization of these ecological networks (Ohlmann et al. (2019) <doi:10.1111/ele.13221>, Gonzalez et al. (2020) <doi:10.1101/2020.04.02.021691>, Miele et al. (2021) submitted).
Imports stats, igraph, rdiversity, Matrix.utils, blockmodels, bipartite, Rcpp
SystemRequirements GNU GSL
Depends R (>= 3.5.0)
LinkingTo Rcpp, RcppEigen, RcppGSL
License GPL-3
URL https://plmlab.math.cnrs.fr/econetproject/econetwork
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-03-02 23:00:06 UTC

R topics documented:
econetwork-package .................................................. 2
cpness ................................................................. 3
disPairwise ........................................................... 4
Description

A collection of advanced tools, methods and models specifically designed for analyzing different types of ecological networks.

Details

econetwork is designed for analyzing different types of ecological networks - especially antagonistic (food webs, host-parasite), mutualistic (plant-pollinator, plant-fungus, etc.) and competitive networks, as well as their variability in time and space. Statistical models are developed to describe and understand the mechanisms that determine species interactions, and to decipher the organization of these (multi-layer) ecological networks.

Author(s)

Authors: Stephane Dray, Catherine Matias, Vincent Miele, Marc Ohlmann, Wilfried Thuiller Maintainer: Vincent Miele <vincent.miele@univ-lyon1.fr>

References


See Also

igraph bipartite
Description

Computation of the cpness measure for a bipartite graph/network

Usage

cpness(web, type=c("automatic","binary","integer","float"), plot=TRUE, fastplot=FALSE)

Arguments

web
A matrix with elements of a set (e.g., plants) as rows, elements of a second set (e.g., pollinators) as columns and number of interactions as entries. Species names can be indicated in the row or column names.

type
Type of matrix. This should be (an unambiguous abbreviation of) one of automatic (default), binary, integer or float. With automatic (default), the type of matrix is automatically deduced inside the function. Using binary, the matrix is binarized in any case. See the "Details" section.

plot
Plot the matrix reordered according to the core-periphery partitioning. TRUE by default.

fastplot
If TRUE, the matrix plot is performed using the fast image function instead of the visweb function from the bipartite package, without species names. Only for large matrices. FALSE by default.

Details

In a matrix displaying a core-periphery structure, there is a species ordering (i.e. an ordering in rows and columns) such that interactions are distributed in an L-shape. This L-shape is composed by four blocks of varying connectance: block C11 represents the core; blocks C12 and C21 include the interactions between core and periphery; block C22 includes the interactions that occur between peripheral species.

This function computes the core-peripheriness (CPness), as CPness=\((E_{11}+E_{12}+E_{21})/E\), where \(E_{ij}\) is the number of interactions (edges) or the sum of weights for each block (\(E_{ij}\) for block \(ij\)) or for the entire network (\(E\)). Here, we rely on a stochastic block model (SBM) to detect the four groups/blocks C11, C12, C21 and C22 when they actually exist. However, the SBM can fail in finding these blocks: in these cases, there is no core-periphery partition and the CPness value is set to NA.

This function can deal with binary and weighted networks with the appropriate statistical distribution (Bernouilli for binary data, Poisson for integer weights, and Gaussian for float weights). Note that it is often advisable to log-transform float data before running the cpness function.
disPairwise

Value

cpness returns an object of class list with the following components:

cpness
The value of the core-peripheriness measure. NA when no core-periphery partition is found.

rowmembership
An integer vector indicating the group to which species in rows are belonging. Group 1 is core and group 2 is periphery, unless there is only a single group.

colmembership
Same as rowmembership for species in columns.

Author(s)

Authors: Vincent Miele
Maintainer: Vincent Miele <vincent.miele@univ-lyon1.fr>

References


Examples

library(bipartite)
data(mosquin1967)
result <- cpness(mosquin1967, type="automatic", plot=TRUE)
print(result)

data(junker2013)
result <- cpness(junker2013, type="automatic", plot=TRUE, fastplot=TRUE)
print(result$cpness)
print(table(result$rowmembership))
print(table(result$colmembership))

disPairwise

Computation of the dissimilarity matrix (pairwise beta-diversity) for a set of networks

Description

Computation of the dissimilarity matrix for a set of networks. Each value of the matrix is the pairwise beta-diversity, computed using Hill numbers. It measures the dissimilarity in terms of groups, links, or probability of links.

Usage

disPairwise(gList, groups=NULL, eta=1,
type=c('P','L','Pi'), abTable=NULL)
Arguments

- **gList**: A list of graph objects of class `igraph`. The nodes must have a name.
- **groups**: A named vector of class character indicating the group to which each node belongs to. The length of groups must correspond to the number of different nodes present in gList. The names `names(groups)` must correspond to the nodes names in gList. If NULL, the groups are the initial nodes.
- **eta**: A positive number that controls the weight given to abundant groups/links. Default value is 1.
- **type**: The type of diversity used to measure dissimilarity. It can be groups diversity ('P'), links diversity ('L') or probability of links diversity ('Pi').
- **abTable**: A matrix of size the number of nodes of the metanetwork times the number of networks. The rownames of this matrix must be the node names of metanetwork and the columns must be in an order corresponding to gList. The element (i,j) of this matrix is the abundance of species i in network j. Importantly, the non-null elements in each column of abTable must correspond to the nodes present in each element of gList.

Value

Return a matrix whose elements are the pairwise dissimilarities.

Author(s)

Authors: Stephane Dray, Vincent Miele, Marc Ohlmann, Wilfried Thuiller
Maintainer: Wilfried Thuiller <wilfried.thuiller@univ-grenoble-alpes.fr>

References


Examples

```r
# Generating a set of Erdos-Renyi graphs and give node names.
library(econetwork)
library(igraph)
nbGraph <- 10
gList <- c()
n <- 57 # number of nodes of each graph
C <- 0.1 # connectance of each graph
for(i in 1:nbGraph){
  graphLocal <- erdos.renyi.game(n, type='gnp', p.or.m=C, directed=TRUE)
  V(graphLocal)$name <- as.character(1:n)
  gList = c(gList,list(graphLocal))
}
# vector that gives the group of each node
groups <- c(rep("a",23),rep("b",34))
```
names(groups) <- as.character(1:57)
# generating random (non-nul) abundances data
abTable <- sapply(1:nbGraph,function(x) rpois(n,1)+1)
rownames(abTable) = unlist(unique(lapply(gList,function(g) V(g)$name)))

# Dissimilarity matrices based on links beta-diversity
# at a node level
disPairwise(gList, type = 'L')
# at a node level while taking into account node abundances
disPairwise(gList, type = 'L', abTable = abTable)
# at a group level
disPairwise(gList, groups, type = 'L')
# at a group level while taking into account node abundances
# disPairwise(gList, groups, type = 'L', abTable = abTable)

---

**divPartition**

*Partitionning network diversity in alpha, beta and gamma diversity*

**Description**

This function computes alpha, beta and gamma diversity of a list of networks. It measures either group, links, or probability of links diversity.

**Usage**

```r
divPartition(gList, groups, eta=1, framework=c('RLC','Chao'),
            type=c('P','L','Pi'), abTable=NULL)
```

**Arguments**

- `gList`: A list of graph objects of class igraph.
- `groups`: A named vector of class character indicating the group to which each node belongs to. The length of groups must correspond to the number of different nodes present in gList. The names names(groups) must correspond to the nodes names in gList. If NULL, the groups are the initial nodes.
- `eta`: A positive number that controls the weight given to abundant groups/links. Default value is 1.
- `framework`: The framework used to partitionate diversity, either Reeve Leinster Cobbold ('RLC') or Chao ('Chao')
- `type`: The type of diversity to measure and partitionate. It can be groups diversity ('P'), link diversity ('L') or probability of link diversity ('Pi').
- `abTable`: A matrix of size the number of nodes of the metanetwork times the number of networks. The rownames of this matrix must be the node names of metanetwork and the columns must be in an order corresponding to gList. The element (i,j) of this matrix is the abundance of species i in network j. Importantly, the non-nul elements in each column of abTable must correspond to the nodes present in each element of gList.
**divPartition**

**Value**

Returns a list the following components:

- **mAlpha**: The mean value of alpha-diversity across all networks.
- **Alphas**: A vector of numeric containing the local alpha-diversities (i.e., the alpha-diversity value for each network).
- **Beta**: The value of the overall beta-diversity
- **Gamma**: The value of the gamma-diversity

**Author(s)**

Authors: Stephane Dray, Vincent Miele, Marc Ohlmann, Wilfried Thuiller
Maintainer: Wilfried Thuiller <wilfried.thuiller@univ-grenoble-alpes.fr>

**References**


**Examples**

```r
# Generating a set of Erdos-Renyi graphs and give node names.
library(econetnetwork)
library(igraph)
nbGraph <- 10
gList <- c()
n <- 57 # number of nodes of each graph
C <- 0.1 # connectance of each graph
for(i in 1:nbGraph){
  graphLocal <- erdos.renyi.game(n, type='gnp', p.or.m=C, directed=TRUE)
  V(graphLocal)$name <- as.character(1:n)
  glist = c(glist,list(graphLocal))
}

# vector that gives the group of each node
groups <- c(rep("a",23),rep("b",34))
names(groups) <- as.character(1:n)

# generating random (non-nul) abundances data
abTable <- sapply(1:nbGraph,function(x) rpois(n,1)+1)
rownames(abTable) = unlist(unique(lapply(gList,function(g) V(g)$name)))

# Diversities in link abundances
# at a node level
divPartition(gList, framework='Chao', type = 'L')
# at a node level while taking into account node abundances
divPartition(gList, framework='Chao', type = 'L', abTable = abTable)
# at a group level
divPartition(gList, framework='Chao', groups, type = 'L')
# at a group level while taking into account node abundances
divPartition(gList, framework='Chao', groups, type = 'L', abTable = abTable)
```
elgrin  

Teasing apart the effects of environmental variations and biotic interactions on species distributions

Description

ELGRIN jointly models the presence and absence of all species in a given area in function of environmental covariates and the topological structure of the known metaweb

Usage

elgrin(presence, metaweb, environmentvar, ncores=1)

Arguments

- **presence**: A matrix with presence/absence of species (rows) at different locations (columns), coded in 0/1 format (1 for presence).
- **metaweb**: A matrix or a igraph object. This represents the metaweb/metanetwork of interactions as an adjacency matrix or as a graph respectively.
- **environmentvar**: A matrix with environmental variables (columns) at different locations (rows).
- **ncores**: The number of cores to use, i.e. at most how many threads will run simultaneously. Default is 1 (no parallelization).

Details

ELGRIN (in reference to Charles Elton and Joseph Grinnell) can handle the effects of both environmental factors and known ecological interactions (aka a metanetwork) on species distributions. We rely on Markov random fields, a family of flexible models that can handle dependencies between variables using a graph.

Value

elgrin returns an object of class list with the following components:

- **a**: A vector of species prevalence.
- **al**: A vector of richness capacity (or expected number of species) at each location.
- **b**: A matrix of linear environmental parameters.
- **c**: A matrix of quadratic environmental parameters.
- **betaPres**: A vector of co-presence strength (or avoidance when <0) at each location.
- **betaAbs**: A vector of co-absence strength (or avoidance when <0) at each location.

Author(s)

Authors: Vincent Miele  
Maintainer: Vincent Miele <vincent.miele@univ-lyon1.fr>
getMetaweb

References
Vincent Miele, Catherine Matias, Marc Ohlmann, Giovanni Poggiato, Stéphane Dray and Wilfried Thuiller, Quantifying the overall effect of biotic interactions on species communities along environmental gradients, submitted.

Examples
```r
## Not run: data(virtualComCompetition)
vcom <- virtualComCompetition
result <- elgrin(vcom$presence, vcom$metaweb, vcom$environmentvar)

## checking results
str(result)
plot(result$betaPres, result$betaAbs, xlab="Co-presence", ylab="Co-absence")
abline(h=0, col=2)
abline(v=0, col=2)

## End(Not run)
```

getMetaweb

Get metaweb

Description
Computation of the binary metaweb from a list of graph

Usage
`getMetaweb(gList)`

Arguments

- `gList` A list of graph objects of class igraph.

Details
This function computes the metaweb from a list of graph. It computes the union (in the sense of graph theory) of the set of graphs.

Value
`getMetaweb` returns an object of class igraph.

Author(s)
Authors: Stephane Dray, Vincent Miele, Marc Ohlmann, Wilfried Thuiller
Maintainer: Wilfried Thuiller <wilfried.thuiller@univ-grenoble-alpes.fr>
References


Examples

```r
# Generating a set of Erdos-Renyi graphs
library(igraph)

gL <- c()
for(i in 1:4){
  graphLocal <- erdos.renyi.game(60, type = 'gnp', p.or.m=0.1, directed=TRUE)
  V(graphLocal)$name <- as.character(1:60)
  gL <- c(gL, list(graphLocal))
}
names(gL) <- c("A","B","C","D")

## building the metaweb
graphMetaweb <- getMetaweb(gL)
```

**virtualComCompetition**  *Dataset simulated with communities whose composition is driven simultaneously by biotic (competition) and abiotic environmental effects.*

Description

50 species in interaction in a metaweb, with presence/absence data for 400 locations and a single environmental variable.

Usage

data(virtualComCompetition)

Format

A list containing three objects presence, metaweb and environmentvar.
Index

* diversity
econetwork-package, 2
* ecology
econetwork-package, 2
* network
econetwork-package, 2
* package
econetwork-package, 2

bipartite, 2

cpnness, 3

disPairwise, 4
divPartition, 6

econetwork (econetwork-package), 2
econetwork-package, 2
elgrin, 8

gMetaweb, 9

igraph, 2

virtualComCompetition, 10