Package 'complexNet'

July 22, 2025

Type Package

Title Complex Network Generation

Version 0.2.0

Description Providing a set of functions to easily generate and iterate complex networks.
The functions can be used to generate realistic networks with a wide range of different clustering, density, and average path length.
For more information consult research articles by Amiyaal Ilany and Erol Akcay (2016) <doi:10.1093/icb/icw068> and Ilany and Erol Akcay (2016) <doi:10.1101/026120>, which have inspired many methods in this package.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

Imports methods

URL https://marcosmolla.github.io/complexNet/,
https://github.com/marcosmolla/complexNet

BugReports https://github.com/marcosmolla/complexNet/issues

Date 2022-11-09 09:00:02 UTC

NeedsCompilation no

Author Marco Smolla [aut, cre] (ORCID:
<https://orcid.org/0000-0001-6367-8765>)

Maintainer Marco Smolla <drsmolla@icloud.com>

Repository CRAN

Date/Publication 2022-11-10 13:10:06 UTC

Contents

`avg_degree_bnr` *Expected average degree of BNR networks*

Description

Calculates the expected average degree of a BNR network (single parent only) based on the approximation by Ilany and Akcay, 2016 (see details).

Usage

```
avg_degree_bnr(n, pb, pn, pr)

## S4 method for signature 'numeric,numeric,numeric,numeric'
avg_degree_bnr(n, pb, pn, pr)
```

Arguments

`n`	Number of nodes in the network
`pb`	Probability to connect to parent (default is 1)
`pn`	Probability to connect to neighbour of parent(s)
`pr`	Probability to connect to individuals that are not connected to

Details

The expected average degree $\bar{d}$ is calculated as

$$\bar{d} = \frac{(N-1)(p_b + (N-2)p_r)}{N - 1 - (N-2)(p_n - p_r)}$$

Value

Returns the expected average degree of a BNR network as a numeric value. This value is an analytic result and not a numeric approximation (compare examples below).

References

Ilany, A., and Akçay, E. (2016). Personality and Social Networks: A Generative Model Approach. Integrative and Comparative Biology, 56(6), 1197–1205. doi:10.1093/icb/icw068

Examples

```
# Expected degree
avg_degree_bnr(n = 100, pb = 1, pn = .2, pr = .02)
# Compare to simulated network with identical parameters
adjm <- make_bnr(n = 100, np = c(0,0), pb = 1, pn = .2, pr = .02)
mean(adjm) * 100
```

`init_graph` *Initialising a random graph*

Description

init_graph takes number of nodes (n) and average degree (deg) to generate a random graph.

Usage

```
init_graph(n, deg)

## S4 method for signature 'numeric,numeric'
init_graph(n, deg)
```

Arguments

n	Number of nodes in the network
deg	Average degree in the network

Value

Returns an unweighted (binary) adjacency matrix, where each cell represents the presence (1) or absence (0) of an interaction between the row and the column individual.

Examples

```
init_graph(n = 10, deg = 4)
```

`iterate_bnr` *Iterating a bnr network*

Description

Iterating a bnr network

Usage

```
iterate_bnr(adjm, np, pb, pn, pr)

## S4 method for signature 'matrix,numeric,numeric,numeric,numeric'
iterate_bnr(adjm, np, pb, pn, pr)
```

Arguments

adjm	Adjacency matrix
np	numeric vector setting ids for the newborn (i.e. which individual will be replaced with a new one) and a parent(s). Length 2 or 3. If you want to randomly select an id for the newborn (first value) and parents (second and third value), simply use c(0,0) or c(0,0,0). For one parent, the focal individual connects to this parent with probability pb. For two parent values, the individual connects to two parents each with probability pb.
pb	Probability to connect to parent. Default is 1.
pn	Probability to connect to neighbour of parent(s)
pr	Probability to connect to individuals that are not connected to the parent

Details

If you just want to iterate the graph you can use `np=c(0,0)` or `np=c(0,0,0)`. However, the function does not return the ids of the newborn and the parent(s). If you want to keep track of the ids that are changed, you should provide these as an input to the function.

Value

Returns an iterated version of the supplied adjacency matrix as a numeric matrix.

Examples

```
# Set up linking parameters:
pb <- 1
pn <- 0.2
pr <- 0.01
# Generate a network based on these parameters
adjm_t0 <- make_bnr(n = 100, np=c(0,0), pb = pb, pn = pn, pr = pr)
# Iterate the network
adjm_t1 <- iterate_bnr(adjm = adjm_t0, np=c(0,0), pb = pb, pn = pn, pr = pr)
```

iterate_kp *Iterating a kp network*

Description

Iterating a kp network

Usage

```
iterate_kp(adjm, np, pb, k, p)

## S4 method for signature 'matrix,numeric,numeric,numeric,numeric'
iterate_kp(adjm, np, pb, k, p)
```

Arguments

adjm	Adjacency matrix
np	numeric vector setting ids for the newborn (i.e. which individual will be replaced with a new one) and a parent(s). Length 2 or 3. If you want to randomly select an id for the newborn (first value) and parents (second and third value), simply use c(0,0) or c(0,0,0). For one parent, the focal individual connects to this parent with probability pb. For two parent values, the individual connects to two parents each with probability pb.
pb	Probability to connect to parent. Default is 1.
k	Degree (number of connections a new individual will form)
p	Maximum proportion of k that will be connections to neighbours of the parent. The complimentary k*(1-p) connections will be formed with random other individuals

Details

If you just want to iterate the graph you can use `np=c(0,0)` or `np=c(0,0,0)`. However, the function does not return the ids of the newborn and the parent(s). If you want to keep track of the ids that are changed, you should provide these as an input to the function.

Value

Returns an iterated version of the supplied adjacency matrix as a numeric matrix.

Examples

```
# Set up linking parameters:
pb <- 1
k <- 4
p <- 0.2
# Generate a network based on these parameters
adjm_t0 <- make_kp(n = 100, np=c(0,0), pb = pb, k = k, p = p)
```

```
# Iterate the network
adjm_t1 <- iterate_kp(adjm = adjm_t0, np=c(0,0), pb = pb, k = k, p = p)
```

---

make_bnr *Generating a Pb, Pn, Pr network*

Description

This function takes adj.matrix (ADJM), probabilities to connect to parent(s), neighbours, and randoms (PB, PN, PR), the index of the parent (if NULL, default, NPARENT number of individuals are randomly chosen as parent), number of parents (NPARENT, default is 1).

Usage

```
make_bnr(n, np, pb, pn, pr)

## S4 method for signature 'numeric,numeric,numeric,numeric,numeric'
make_bnr(n, np, pb, pn, pr)

## S4 method for signature 'numeric,numeric,missing,numeric,numeric'
make_bnr(n, np, pb, pn, pr)
```

Arguments

n	Number of vertices (population size)
np	numeric vector setting ids for the newborn (i.e. which individual will be replaced with a new one) and a parent(s). Length 2 or 3. If you want to randomly select an id for the newborn (first value) and parents (second and third value), simply use c(0,0) or c(0,0,0).
pb	Probability to connect to parent (default is 1)
pn	Probability to connect to neighbour of parent(s)
pr	Probability to connect to individuals that are not connected to the parent

Details

It is important to note that, although all three parameters (PB, PN, PR) are probabilities, i.e. values between 0 and 1, the same value (say 0.2) means something different for each of them. This is because, PB is the probability to connect to the parent(s), i.e. 1 or two individuals. In contrast, PN and PR are the probabilities to connect to neighbours of the parent(s) or to random other individuals. In the case of a small social neighbourhood of the parent(s) a PR of 0.2 would mean to connect to a large amount of individuals in the remaining network. Therefore, it is important to keep in mind that the value of both (or all three) values is important and not the individual one in isolation.

Value

Returns an unweighted (binary) adjacency matrix, where each cell represents the presence (1) or absence (0) of an interaction between the row and the column individual.

Examples

```
make_bnr(n = 10, np = c(0,0), pb = 1, pn = .2, pr = .01)
```

---

make_kp *Generating a kp network*

Description

This function ...

Usage

```
make_kp(n, np, pb, k, p)

## S4 method for signature 'numeric,numeric,numeric,numeric,numeric'
make_kp(n, np, pb, k, p)

## S4 method for signature 'numeric,numeric,missing,numeric,numeric'
make_kp(n, np, pb, k, p)
```

Arguments

n	Number of vertices (population size)
np	numeric vector setting ids for the newborn (i.e. which individual will be replaced with a new one) and a parent(s). Length 2 or 3. If you want to randomly select an id for the newborn (first value) and parents (second and third value), simply use c(0,0) or c(0,0,0).
pb	Probability to connect to parent (default is 1)
k	Degree (number of connections a new individual will form)
p	Maximum proportion of k that will be connections to neighbours of the parent. The complimentary k*(1-p) connections will be formed with random other individuals

Details

It is important to note that ... P is a maximum value, say an individual wants to have 10 connections and P=0.5, i.e. it wants 5 connections to the neighbours of its parent but the parent only has 4 then it will only inherit those 4.

Value

Returns an unweighed (binary) adjacency matrix, where each cell represents the presence (1) or absence (0) of an interaction between the row and the column individual.

Examples

```
make_kp(n = 10, np = c(0,0), pb = 1, k = 4, p = .5)
```

Index