

# Package ‘miaSim’

January 19, 2022

**Type** Package

**Version** 1.1.0

**Title** Microbiome Data Simulation

**Description** Microbiome time series simulation with generalized Lotka-Volterra model, Self-Organized Instability (SOI), and other models. Hubbell's Neutral model is used to determine the abundance matrix. The resulting abundance matrix is applied to SummarizedExperiment or TreeSummarizedExperiment objects.

**License** Artistic-2.0 | file LICENSE

**biocViews** Microbiome, Software, Sequencing, DNaseq, ATACSeq, Coverage, Network

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.1.2

**Depends** SummarizedExperiment

**Imports** deSolve, stats, powerLaw

**Suggests** rmarkdown, knitr, BiocStyle, testthat

**URL** <https://github.com/microbiome/miaSim>

**BugReports** <https://github.com/microbiome/miaSim/issues>

**Roxygen** list(markdown = TRUE)

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/miaSim>

**git\_branch** master

**git\_last\_commit** 6e819af

**git\_last\_commit\_date** 2021-10-26

**Date/Publication** 2022-01-19

**Author** Karoline Faust [aut],  
Yu Gao [aut],  
Emma Gheysen [aut],  
Daniel Rios Garza [aut],

Yagmur Simsek [cre, aut],  
 Leo Lahti [aut] (<<https://orcid.org/0000-0001-5537-637X>>)

**Maintainer** Yagmur Simsek <yagmur.simsek@hsrw.org>

## R topics documented:

powerlawA . . . . .	2
randomA . . . . .	3
simulateGLV . . . . .	4
simulateHubbell . . . . .	5
simulateSOI . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

---

powerlawA	<i>Interaction matrix with Power-Law network adjacency matrix</i>
-----------	---

---

### Description

Where N is the an Interspecific Interaction matrix with values drawn from a normal distribution H the interaction strength heterogeneity drawn from a power-law distribution with the parameter alpha, and G the adjacency matrix of with out-degree that reflects the heterogeneity of the powerlaw. A scaling factor s may be used to constrain the values of the interaction matrix to be within a desired range. Diagonal elements of A are defined by the parameter d.

### Usage

```
powerlawA(n.species, alpha = 3, stdev = 1, s = 0.1, d = -1, symmetric = FALSE)
```

### Arguments

n.species	integer: the number of species
alpha	numeric: the power-law distribution parameter. Should be > 1. (default: alpha = 3.0) Larger values will give lower interaction strength heterogeneity, whereas values closer to 1 give strong heterogeneity in interaction strengths between the species. In other words, values of alpha close to 1 will give Strongly Interacting Species (SIS).
stdev	numeric: the standard deviation parameter of the normal distribution with mean 0 from which the elements of the nominal interspecific interaction matrix N are drawn. (default: stdev = 1)
s	numeric: scaling parameter with which the final global interaction matrix A is multiplied. (default: s = 0.1)
d	numeric: diagonal values, indicating self-interactions (use negative values for stability). (default: s = 1.0)
symmetric	logical: return a symmetric interaction matrix (default: symmetric = FALSE)

**Value**

The interaction matrix A with n rows and n columns.

**References**

Gibson TE, Bashan A, Cao HT, Weiss ST, Liu YY (2016) On the Origins and Control of Community Types in the Human Microbiome. PLOS Computational Biology 12(2): e1004688. <https://doi.org/10.1371/journal.pcbi.1004688>

**Examples**

```
# Low interaction heterogeneity
A_low <- powerlawA(n.species = 10, alpha = 3)
# Strong interaction heterogeneity
A_strong <- powerlawA(n.species = 10, alpha = 1.01)
```

---

randomA

*Generate random uniform interaction matrix*

---

**Description**

Generate random simplified interaction matrix from a uniform distribution.

**Usage**

```
randomA(
  n.species,
  d = -0.5,
  min.strength = -0.5,
  max.strength = 0.5,
  connectance = 0.02,
  symmetric = FALSE
)
```

**Arguments**

n.species	integer number of species
d	numeric diagonal values (should be negative) (default: d = -0.5)
min.strength	numeric minimal off-diagonal interaction strength (default: min.strength = -0.5)
max.strength	numeric maximal off-diagonal interaction strength (default: max.strength = 0.5)
connectance	numeric (interaction probability) (default: connectance = 0.02)
symmetric	logical return a symmetric interaction matrix (default: symmetric = FALSE)

**Value**

randomA returns a matrix A with dimensions (n.species x n.species)

**Examples**

```
high_inter_A <- randomA(n.species = 10, d = -0.4, min.strength = -0.8,
                       max.strength = 0.8, connectance = 0.5)

low_inter_A <- randomA(n.species = 10, connectance = 0.01)
```

---

simulateGLV

*Generalized Lotka-Volterra (gLV) simulation*


---

**Description**

Simulates time series with the generalized Lotka-Volterra model and forms a [SummarizedExperiment](#) object.

**Usage**

```
simulateGLV(
  n.species,
  A,
  x = runif(n.species),
  b = runif(n.species),
  t.end = 1000,
  norm = FALSE,
  ...
)
```

**Arguments**

n.species	Integer: number of species
A	interaction matrix
x	Numeric: initial abundances
b	Numeric: growth rates
t.end	Numeric: simulation end time (default: t.end = 1000)
norm	Logical scalar: returns normalised abundances (proportions in each generation) (default: norm = FALSE)
...	additional arguments that can be called from miaSim::tDyn

**Details**

Simulates a community time series using the generalized Lotka-Volterra model, defined as  $dx/dt = x(b+Ax)$ , where  $x$  is the vector of species abundances,  $\text{diag}(x)$  is a diagonal matrix with the diagonal values set to  $x$ .  $A$  is the interaction matrix and  $b$  is the vector of growth rates.

The resulting abundance matrix model is used to construct [SummarizedExperiment](#) object.

**Value**

simulateGLV returns a [SummarizedExperiment](#) object containing abundance matrix

**Examples**

```
row_data <- data.frame(Kingdom = "Animalia",
                      Phylum = rep(c("Chordata", "Echinodermata"), c(500, 500)),
                      Class = rep(c("Mammalia", "Asteroidea"), each = 500),
                      ASV = paste0("X", seq_len(1000)),
                      row.names = rownames(paste0("species", seq_len(1000))),
                      stringsAsFactors = FALSE)

row_data <- t(row_data)

col_data <- DataFrame(sampleID = seq_len(1001),
                     time = as.Date(1000, origin = "2000-01-01"),
                     row.names = colnames(paste0("sample", seq_len(1001))))

A <- miaSim::powerlawA(4, alpha = 1.01)

SEobject <- simulateGLV(n.species = 4, A, t.end = 1000)
rowData(SEobject) <- row_data
colData(SEobject) <- col_data
```

---

simulateHubbell

*Hubbell's neutral model simulation*

---

**Description**

Neutral species abundances simulation according to the Hubbell model.

**Usage**

```
simulateHubbell(
  n.species,
  M,
  I = 1000,
  d = 10,
  m = 0.02,
  tskip = 0,
  tend,
  norm = FALSE
)
```

**Arguments**

n.species	Integer: the amount of different species initially in the local community
M	Integer: amount of different species in the metacommunity, including those of the local community
I	Integer: fixed amount of individuals in the local community (default: I = 1000)
d	Integer: fixed amount of deaths of local community individuals in each generation (default: d = 10)
m	Numeric: immigration rate: the probability that a death in the local community is replaced by a migrant of the metacommunity rather than by the birth of a local community member (default: m = 0.02)
tskip	Integer: number of generations that should not be included in the outputted species abundance matrix. (default: tskip = 0)
tend	Integer: number of simulations to be simulated
norm	Logical: whether the time series should be returned with the abundances as proportions (norm = TRUE) or the raw counts (default: norm = FALSE)

**Value**

simulateHubbell returns a [SummarizedExperiment](#) class object containing matrix with species abundance as rows and time points as columns

**References**

Rosindell, James et al. "The unified neutral theory of biodiversity and biogeography at age ten." Trends in ecology & evolution vol. 26,7 (2011).

**Examples**

```
colData <- DataFrame(sampleID = c(seq_len(100)),
                    time = as.Date(100, origin = "2000-01-01"))

rowData <- data.frame(Kingdom = "Animalia",
                    Phylum = rep(c("Platyhelminthes", "Mollusca"), c(50, 50)),
                    Class = rep(c("Turbellaria", "Polyplacophora"), each = 50),
                    ASV1 = paste0("D", seq_len(100)),
                    ASV2 = paste0("E", seq_len(100)),
                    ASV3 = paste0("F", seq_len(100)),
                    ASV4 = paste0("G", seq_len(100)),
                    ASV5 = paste0("H", seq_len(100)),
                    ASV6 = paste0("J", seq_len(100)),
                    ASV7 = paste0("K", seq_len(100)),
                    row.names = rownames(paste0("species", seq_len(10))),
                    stringsAsFactors = FALSE)

rowData <- t(rowData)

ExampleHubbell <- simulateHubbell(n.species = 8, M = 10, I = 1000, d = 50,
                                m = 0.02, tend = 100)
```



# Index

powerlawA, [2](#)

randomA, [3](#)

simulateGLV, [4](#)

simulateHubbell, [5](#)

simulateNeutral (simulateHubbell), [5](#)

simulateSOI, [7](#)

SummarizedExperiment, [4–7](#)