Package: ibm (via r-universe)

September 11, 2024

Version 0.3.0

Title Individual Based Models in R

Description Implementation of some (simple) Individual Based Models and methods to create new ones, particularly for population dynamics models (reproduction, mortality and movement). The basic operations for the simulations are implemented in Rcpp for speed.

Depends R (>= 2.15)

Imports Rcpp (>= 0.11.5), stats, graphics

License GPL-2

URL https://roliveros-ramos.github.io/ibm/

BugReports https://github.com/roliveros-ramos/ibm/issues

LinkingTo Rcpp

RoxygenNote 7.2.3

Encoding UTF-8

Repository https://roliveros-ramos.r-universe.dev

RemoteUrl https://github.com/roliveros-ramos/ibm

RemoteRef HEAD

RemoteSha fdd0b09dba6ab85122bb3426804d66e3ccc0d1f2

Contents

`ibm-package` *Individual based models in R*

Description

Implementation of some (simple) Individual Based Models and methods to create new ones, particularly for population dynamics models (reproduction, mortality and movement). The basic operations for the simulations are implemented in Rcpp for speed.

Author(s)

Ricardo Oliveros-Ramos <ricardo.oliveros@gmail.com>

Examples

```
## Not run:
set.seed(880820)
par = list(alpha=5e-4, beta=5e-4, r=0.1, m=0.05, D=list(N=8e-5, P=8e-5),
L=list(N=0.2, P=0.2))
N0 = with(par, m/(2*beta*L$P))
P0 = with(par, r/(2*alpha*L$N))
par$initial = list(N=round(N0), P=round(P0))
sim = localLotkaVolterra(par, T=240, replicates=100, maxpop = 1e4)
plot(sim)

## End(Not run)
```

`boundaries` *Spatial boundary restrictions*

Description

Set spatial restrictions to the domain.

Usage

```
boundaries(x, ...)
```

Arguments

`x`	The positions of the particles.
`...`	Additional arguments for different methods.

Details

Boundaries is a generic and methods can be written. The default applies simmetric boundaries (dynamics over a torus) or reflexive barriers.

`diffusion` *Brownian diffusion of a set of particles*

Description

This funtions performs a brownian difussion over a set of particles. The dimension is automatically calculated from the number of columns of the object.

Usage

```
diffusion(object, sd, ...)
```

Arguments

`object`	The positions of the particles, dimension is taken from the number of columns or assumed to be 1 is no columns.
`sd`	Standard deviation for the gaussian jump, for dynamics models should be set proportional to `sqrt(dt)`.
`...`	Additional arguments for different methods.

Details

This functions apply a brownian diffusion to a set of point coordinates.

`localLotkaVolterra` *Lotka-Volterra with local predation interactions*

Description

This function simulates several trajectories for a Lotka-Volterra model with local predation interactions as decribed in Brigatti et al. (2009).

Usage

```
localLotkaVolterra(
  par,
  T,
  replicates = 1,
  dim = 1,
  periodic = TRUE,
  spatial = FALSE,
  verbose = FALSE,
  maxpop = 1e+06,
  fill = NA,
  probs = seq(0, 1, 0.01)
)
```

Arguments

`par`	A list containing the parameters to run the model, currently the growth rate of prey (r), the mortality rate of predator (l), predation interaction parameters (alpha and beta), diffusion rates (D), diameters of local interaction (L) and initial population size (initial). For D, L and initial population, a list with two values (named N and P) is required.
`T`	Time horizon, number of time steps to be simulated.
`replicates`	Number of replicates (trajectories) to be simulated.
`dim`	Spatial dimension for the space. Can be 1, 2 or 3.
`periodic`	Spatial boundary conditions. If `periodic` is set to `TRUE`, the space is a torus. If set to `FALSE`, the boundaries are reflective.
`spatial`	Boolean, should spatial outputs (position of individuals) to be saved?
`verbose`	Boolean, to print population sizes by step?
`maxpop`	Maximum population size. If predator or prey population size
`fill`	Value to initially fill the population arrays. Default to NA, 0 is an option too.
`probs`	Vector of probabilities to compute quantiles of the predator-prey interactions. get bigger, the simulation ends.

Value

A list with the following elements:

`N`	A matrix with prey population sizes by time (rows) and replicates (columns)
`P`	A matrix with predator population sizes by time (rows) and replicates (columns)
`pop`	Prey and predator positions by time, if `spatial` is `TRUE`

Author(s)

Ricardo Oliveros–Ramos

References

Brigatti et al. 2009.

Examples

```
## Not run:
set.seed(880820)
par = list(alpha=5e-4, gamma=0.5, r=0.1, m=0.05, D=list(N=8e-5, P=8e-5),
L=list(N=0.2, P=0.2))
N0 = with(par, m/(2*gamma*alpha*L$P))
P0 = with(par, r/(2*alpha*L$N))
par$initial = list(N=round(N0), P=round(P0))
sim = localLotkaVolterra(par, T=240, replicates=100, maxpop = 1e4)
plot(sim)

## End(Not run)
```

`mortality` *Mortality Process*

Description

This functions performs the 'mortality' process over an object, decreasing the number of individuals. It is a generic, S3 methods can be specified for a particular specification of the population.

Usage

```
mortality(object, rates, ...)
```

Arguments

`object`	The population object, containing the information about individuals.
`rates`	The mortality rate or rates.
`...`	Additional arguments for different methods.

Details

The rate can be a single value or a value for each individual calculated externally. No recycling is allowed.

`reproduction` *Reproduction Process*

Description

This functions performs the 'reproduction' process over an object, increasing the number of individuals. It is a generic, S3 methods can be specified for a particular specification of the population.

Usage

```
reproduction(object, rates, ...)
```

Arguments

`object`	The population object, containing the information about individuals.
`rates`	The reproduction rate or rates.
`...`	Additional arguments for different methods.

Details

The rate can be a single value or a value for each individual calculated externally. No recycling is allowed.

Index