Package ‘briskaR’

June 7, 2018

Type Package

Encoding UTF-8

Title Biological Risk Assessment

Version 0.1.2

Date 2018-06-05

Author Emily Walker [aut],
Jean-Francois Rey [aut, cre],
Melen Leclerc [aut],
Samuel Soubeyrand [ctb],
Marc Bourotte [ctb]

Maintainer Jean-Francois Rey `<jean-francois.rey@inra.fr>`

Description A spatio-temporal exposure-hazard model for assessing biological risk and impact. The model is based on stochastic geometry for describing the landscape and the exposed individuals, a dispersal kernel for the dissemination of contaminants and an ecotoxicological equation.
Walker E, Leclerc M, Rey JF, Beaudouin R, Soubeyrand S, and Messean A, (2017),
A Spatio-Temporal Exposure-Hazard Model for Assessing Biological Risk and Impact,
Risk Analysis, <doi:10.1111/risa.12941>.

URL `https://gitlab.paca.inra.fr/biosp/briskaR`

BugReports `https://gitlab.paca.inra.fr/biosp/briskaR/issues`

License GPL (>= 2) | file LICENSE

LazyData True

BuildVignettes True

VignetteBuilder knitr

Depends methods, grDevices (>= 3.0.0), graphics (>= 3.0.0), R (>= 3.0.0)

Imports sp (>= 1.0-17), stats (>= 3.0.2), rgeos(>= 0.3), fields(>= 7.1), MASS(>= 7.3.29), deldir(>= 0.1), pracma(>= 1.8.3), raster(>= 2.3.0), fftwtools(>= 0.9.6), mvtnorm(>= 1.0.2), rgdal (>= 0.9)

Suggests knitr(>= 1.11), rmarkdown(>= 0.8.1), testthat(>= 1.0.0)

Collate '0briskaR.R' 'Class-Individuals.R' 'Class-Landscape.R' 'Individuals-Methods.R' 'simulPrecip.R' 'toxicIntensity.R' 'Landscape-Methods.R' 'Landscape-Individuals-Methods.R' 'briskaR.R' 'pollen-data.R' 'ecoToxic.R' 'demo.R'

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-07 19:17:56 UTC

R topics documented:

`briskaR-package` *Biological Risk Assessment*

Description

A spatio-temporal exposure-hazard model for assessing biological risk and impact. The model is based on stochastic geometry for describing the landscape and the exposed individuals, a dispersal kernel for the dissemination of contaminants and an ecotoxicological equation.

Details

Package:	briskaR
Type:	Package
Version:	0.1.2
Date:	2018-06-05
License:	GPL (>=2)

The briskaR package contains functions and methods for quantifying spatio-temporal variation in contamination risk around known polygon sources of contaminants, and quantifies the impact of the contaminants on the surrounding population of individuals which are located in habitat areas and are susceptible to the contaminants.

The package implements an spatio-temporal exposure-hazard model based on (i) tools of stochastic geometry (marked polygon and point processes) for structuring the landscape and describing the location of exposed individuals, (ii) a method based on a dispersal kernel describing the dissemination of contaminant particles from polygon sources, and (iii) an ecotoxicological equation describing how contaminants affect individuals of the exposed population.

Author(s)

Emily Walker <`emily.walker@inra.fr`>

Jean-Francois Rey <`jean-francois.rey@inra.fr`>

Melen Leclerc <`melen.leclerc@inra.fr`>

Samuel Soubeyrand <`Samuel.Soubeyrand@inra.fr`>

Marc Bourotte <`marc.bourotte@inra.fr`>

Maintainer: Jean-Francois REY <`jean-francois.rey@inra.fr`>

See Also

`demo.pollen.run`

Examples

```
## Not run:
## Run a simulation
library("briskaR")
demo.pollen.run()

## End(Not run)
```

---

briskaRLoadInternProjection *Load an internal working projection PROJ.4*

Description

Will load a projection as internal package working projection

Usage

```
briskaRSetInternProjection(proj = LAMBERT_93)
```

Arguments

`proj`	A character string of projection arguments, must be in the PROJ.4 documentation

---

create.pollen.sources *Pollen sources emission simulation*

Description

Simulate pollen sources emission for maize crop. The proportion of plants emitting pollen per day (during 12 days) was observed by Frédérique Angevin (Angevin et al. 2008).

Usage

```
create.pollen.sources(nbOfSource=200, numberOfDay=60,
density=runif(1,7,11), pollen=rgamma(1,shape=1.6,scale=1/(2*10^-7)))
```

Arguments

`nbOfSource`	Number of source fields
`numberOfDay`	Number of time units (e.g. days) including the pollen emission (for simulation, default is time.max). Minimal value is 12 days.
`density`	Plant density (number of plant by squared meter)
`pollen`	Pollen production (number of grains by plant)

Value

A matrix indexed by sources ID (in rows) and by time (in columns) whose rows give the values of pollen emission (number of grains) for every source.

Examples

```
# simulation of an emission matrix for 20 emitting sources and a emission period of 15 days:
create.pollen.sources(nbOfSource=20,numberOfDay=15)
# simulation of an emission matrix for 1 emitting source and a emission period of 20 days:
pollen.emission<-create.pollen.sources(nbOfSource=1,numberOfDay=20)
plot(pollen.emission[1,],xlab="time unites",ylab="maize crop emission")
```

`demo.pollen.run` *briskaR package pollen demonstration*

Description

Demonstration of briskaR package on Bt maize pollen (Genetically Modified crop) on non target Lepidoptera larvae.

Usage

```
demo.pollen.run(nb_fields=100,max_size=5000,raster_size=2^10,nb_ind=100)
```

Arguments

`nb_fields`	number of fields (sources and neutrals) in the landscape (default 100)
`max_size`	landscape size (in meter) (default 5000)
`raster_size`	raster size (default 2^10)
`nb_ind`	number of individuals to simulate (default 100)

Details

In this example, 40% of fields are sources (40 fields) and 100 individuals are exposed to toxic pollen. The time simulation of this demo is 30 days.

This demo takes about 2 minutes running, according to the computer.

Value

A list of the simulated Landscape-class object, the Individuals-class object, and the toxicIntensity array [time, x, y]. The map of landscape with toxic intensity and individual states is plotted at time 30 (end of simulation).

Examples

```
## Not run:
# Demo for toxic pollen dispersion on (100) exposed individuals with 100 fields
# in a squared domain of 5000x5000meters.
# The domain is discretized by 1024x1024 pixels (2^10).
demo.pollen.run(nb_fields=100,max_size=5000,raster_size=2^10,nb_ind=100)

## End(Not run)
```

`ecoToxic` *EcoToxicological model method*

Description

Generic method on Landscape and Individuals objects applying ecotoxicological equation.

This method gives internal concentration of contaminants within individuals, from toxic quantities in the environment and individual parameters.

Usage

```
ecoToxic(objectL, objectI, ...)

## S4 method for signature 'Landscape,Individuals'
ecoToxic(objectL, objectI, objectT,
  mintime = 1, maxtime, kin = 0.25, kout = 0.5, deltat = 0.1)
```

Arguments

`objectL`	A Landscape object
`objectI`	An Individuals object
`...`	other parameters
`objectT`	A ToxicIntensityRaster object, a 3D array of Toxic Dispersion over time [t,x,y], first indice is time
`mintime`	Time to start simulation (default = 1)
`maxtime`	Time to end simulation
`kin`	ingestion rate (% of contaminants staying in the body)
`kout`	elimination rate (% of contaminants eliminated from the body)
`deltat`	% of a time unit for the ordinary differential equation (ODE)

Value

An Individuals object with updated internal toxic concentrations

Examples

```
## Not run:
data(maize_65)
# Calculation of toxic internal concentration of individuals :
res <- ecoToxic(alandscape, anindividuals, atoxicintensity,
 mintime=1, maxtime=60, kin=0.25, kout=0.5, deltat=0.1)
# time series of toxic internal concentration for the 1st individual:
plot(alandscape, res, numind=1)
# Simulate pollen dispersion
tox <- toxicIntensity(maize.landscape,maize.emitted_pollen,1,61)
# Simulate ecotoxicological for individuals plot ecotoxicology of individual 1.
ind2<-ecoToxic(maize.landscape,maize.individuals,objectT=tox,maxtime=61)
# draw result for individual 1.
plot(maize.landscape,ind2,objectT=tox,numind=1)

## End(Not run)
```

`getIndividualsLife` *Method to get Individuals Life information*

Description

Get individuals toxic concentration over the simulation time and return life states for individuals.

Usage

```
getIndividualsLife(object, ...)

## S4 method for signature 'Individuals'
getIndividualsLife(object)
```

Arguments

`object`	An Individuals object
`...`	other parameters

Details

If intern concentration overtakes the toxic threshold value is "-2", that means the individual is dead because of higher toxic concentration. Otherwise value is "-1" means the individual is dead in natural way. The value "0" means that the individual is not alive yet.

Value

a matrix indexed by individual ID in rows and by time in columns.

Examples

```
# the fifth first individuals states for each time step
data(maize_65)
life<-getIndividualsLife(maize.individuals)
matplot(1:10,life[1:10,],type='l',col=1:5,xlab="source",ylab="states")
```

`GetInternProjection` *Get the internal working projection PROJ.4*

Description

Will print and return the internal projection of briskaR package

Usage

```
briskaRGetInternProjection()
```

`getSPSources` *Method to get SpatialPolygons Sources from a Landscape*

Description

Method to get SpatialPolygons Sources from a Landscape

Get all polygons of a Landscape object identified as sources

Usage

```
getSPSources(object, ...)

## S4 method for signature 'Landscape'
getSPSources(object)
```

Arguments

`object`	A Landscape object
`...`	options

Value

a SpatialPolygonDataFrame object

Examples

```
## plot sources fields of maize.landscape
data(maize_65)
sources<-getSPSources(maize.landscape)
plot(sources)
```

`Individuals` *Class Individuals*

Description

`Individuals` Class consists of spatio-temporal parameters about exposed populations.

Individual gets coordinates (as SpatialPoints), a date of birth, a life duration, an internal toxic concentration over the time and a toxic threshold (max value of toxic before death).

Each individual in an Individuals object is identified by an ID which is used as index to access attributes in the object.

Details

Objects can be created by calling of the allocator new("Individuals", ...), or (preferred) by calling one of the wrapped functions simulateIndividuals or loadIndividuals.

Slots

`n` individuals Number

`coordinate` individuals coordinates (as SpatialPoints)

`xmin` x-axis left value

`xmax` x-axis right value

`ymin` y-axis bottom value

`ymax` y-axis top value

`dob` Date of birth (as vector)

`life_duration` individuals life duration (as vector)

`intern_toxic` individuals intern toxic concentration in time (as matrix)

`toxic_threshold` individuals max toxic concentration leading to death (as vector)

`mintime` Start simulation time

`maxtime` End simulation time

See Also

`simulateIndividuals` , `loadIndividuals`

`Individuals plot` *Plot method for Individuals-class*

Description

Will plot Individuals spatial positions.

Will plot individuals positions and state at a time of the simulation.

Usage

```
## S4 method for signature 'Individuals,ANY'
plot(x, y, add = F, ..., plot.legend = TRUE)

## S4 method for signature 'Individuals,numeric'
plot(x, y, add = F, ..., plot.legend = TRUE)
```

Arguments

`x`	An Individuals object
`y`	time of the simulation to display individuals
`add`	if True the new plot will overlap an already plot image (default False)
`...`	further graphical parameters (`par`)
`plot.legend`	plot legend (default TRUE)

Details

The states correspond to internal concentration of toxic. "Red" cross means that the individual is dead because of toxic exposition. "Green" cross means that the individual is dead by natural death. "Green" to "Red" points give the gradient of toxic concentration before the threshold.

Examples

```
data(maize_65)
plot(maize.individuals)
data(maize_65)
# individuals locations
plot(maize.individuals)
# individuals at time 61
plot(maize.individuals,61)
#individuals at time 61 with the landscape
plot(maize.landscape,maize.individuals,time=61)
```

`LAMBERT_93` *LAMBERT_93*

Description

SIG projection Lambert_93 references "+init=epsg:2154" PROJ.4

Usage

```
LAMBERT_93
```

Format

An object of class `character` of length 1.

`Landscape` *Class Landscape*

Description

Class `Landscape` defines a landscape

Details

`Landscape` objects can be created by calling of the allocator new("Landscape", ...), or (preferred) by calling to the function simulateLandscape or loadLandscape.

Slots

`thelandscape` A SpatialPolygonsDataFrame

`xmin` Left x-axis coordinate (in meters)

`xmax` Right x-axis coordinate (in meters)

`ymin` Bottom y-axis coordinate (in meters)

`ymax` Top y-axis coordinate (in meters)

`n` Number of fields in the landscape

See Also

simulateLandscape , loadLandscape , loadLandscapeSIG

loadIndividuals *Wrapper function loadIndividuals*

Description

Wrapper function to create an Individuals object using SpatialPoints and data.frame .

The SpatialPoints object and the data.frame have to contain the same number of coordinates and rows.

Usage

```
loadIndividuals(objectL, sp, data, mintime, maxtime)
```

Arguments

`objectL`	a Landscape object
`sp`	a SpatialPoints object (individuals coordinates)
`data`	a data.frame containing individuals attributes. Rows numbers as individuals ID, columns names as dob (date of birth) | life_duration | toxic_threshold
`mintime`	Start simulation time
`maxtime`	End simulation time

Value

an `Individuals` object

Examples

```
library(sp)
data(maize_65)
# simulate 2 individuals coordinates (SpatialPoints object) and data:
coordinates <- SpatialPoints(matrix(c(468232.1,6259993,464848.8,6260483),ncol=2,byrow=TRUE),
proj4string=CRS("+init=epsg:2154"))
df <- data.frame("dob"=c(1,8),"life_duration"=c(20,20),
           "toxic_threshold"=c(15,15),row.names = c(1,2))
# create an Indiviudals object from previous data
ind <- loadIndividuals(objectL=maize.landscape,sp=coordinates,data=df,mintime=1,maxtime=61)
plot(maize.landscape,ind)
```

`loadLandscape` *Wrapper function : loadLandscape*

Description

Wrapper function to create a Landscape object using SpatialPolygons and dataframe. The SpatialPolygons object and the data.frame have to contain the same number of polygons and row (row ID is polygons ID).

Usage

```
loadLandscape(sp, data)
```

Arguments

`sp`	a SpatialPolygons object designing the landscape
`data`	a data.frame containing fields (polygons) information. Row names as fields ID, column names as sources | neutral | receptors (for a given field, the value is 1 for the type of the field (source or neutral or receptor), otherwise 0).

Value

A Landscape object

Examples

```
data(maize_65)
l<-loadLandscape(maize.landscape@thelandscape,maize.landscape@thelandscape@data)
plot(l)
```

`loadLandscapeSIG` *Create a Landscape object from SIG shapefile file*

Description

Create a Landscape object from SIG shapefile. Shapefile has to contain a SpatialPolygonsDataFrame. Data in the data frame contain fields (polygons) information. Row names as fields ID, cols names as sources | neutral | receptors (for a given field, the value is 1 for the type of the field (source or neutral or receptor), otherwise 0).

Usage

```
loadLandscapeSIG(dsn, layer, format = TRUE)
```

Arguments

`dsn`	folder path to the source files
`layer`	file name without extension
`format`	only load data needed Landscpae-class (default TRUE)

Value

A Landscape object

Examples

```
## Not run:
land<-loadLandscapeSIG("/path/to/directory/","fileName")
plot(land)

## End(Not run)
```

`maize.emitted_pollen` *Pollen sources emission from maize crop.*

Description

Pollen emission for the sources fields of the maize.landscape Landscape over 61 days.

Usage

```
data("maize.emitted_pollen")
```

Format

A data frame with 184 observations (sources emission ID's by rownames) following by 61 values (time unit).

Examples

```
data(maize.emitted_pollen)
```

maize.individuals *Maize_65 Individuals repartitions*

Description

`Individuals-class` of 100 individuals in receptors fields of `maize.landscape`.

Format

`Individuals-class`.

It contains 100 random individuals.

Details

It contains the simulation of 100 individuals in receptors fields of `maize.landscape`. All individuals have a life duration of 20 days within the simulation time [1-61]. Individuals @intern_toxic come from ecoToxic function.

Examples

```
data("maize.individuals")
plot(maize.individuals)
```

maize.landscape *Real Landscape maize_65 with attributs*

Description

The Landscape maize_65 with the distribution of neutral, sources, receptors fields.

Format

`Landscpae-class`

Details

It contains 184 sources, 171 neutrals and 105 recpetors fields within the 460 fileds.

Examples

```
data("maize.landscape")
plot(maize.landscape)
```

`maize.proportion_pollen` *Maize Proportion Pollen*

Description

The proportion of plants emitting pollen per day (during 12 days) was observed by Frédérique Angevin (Angevin et al. 2008).

Usage

```
data("maize.proportion_pollen")
```

Format

The format is a vector of 12 days with proportions of plants emitting.

Source

Frédérique Angevin (Angevin et al. 2008).

Examples

```
data(maize.proportion_pollen)
```

`maize.toxicintensity61` *Pollen dispersion at time 61 over the maize.landscape from toxicIntensity*

Description

Matrix of pollen dispersion at time 61 in the `maize.landscape` Landscape from toxicIntensity function.

Format

The format is a matrix 1024x1024 .

Examples

```
data("maize.toxicintensity61")
```

`maize_65` *Real Landscape*

Description

Real Landscape from Hautes-Pyrénées (65) in South-West of France. The Landscape is an area of 5000x5000 meters near Rabastens-de-Bigorre taken in 2012.

Usage

```
data("maize_65")
```

Format

`Landscape` class object.

It contains 315 maize sources fields within 460 fields.

Source

Open data come from https://www.data.gouv.fr/fr/datasets/registre-parcellaire-graphique-2012-contours-des-ilots-culturaux-et-leur-groupe-de-cultures-majorita/

Examples

```
data(maize_65)
```

`plot Landscape-class` *Plot Method for a Landscape-class*

Description

Plot a Landscape object. If `time` and `objecT` (as ToxicIntensityRaster class attribut) is specified, the values of `objectT` at time `time` are draw over the landscape.

Usage

```
## S4 method for signature 'Landscape,ANY'
plot(x, y, add = F, objectT = NULL, time = -1,
  ..., plot.legend = TRUE, plot.legend.raster = TRUE)
```

Arguments

`x`	A Landscape object
`y`	ANY
`add`	boolean to add draw hover plot
`objectT`	Toxic intensity matrix (result of toxicIntensity, default NULL)
`time`	Time selection (default = -1) for Toxic intensity
`...`	default plot par
`plot.legend`	plot legend (default TRUE)
`plot.legend.raster`	plot raster legend (default TRUE)

Examples

```
## Not run:
# plot a Landscape object
data(maize_65)
plot(maize.landscape)
# plot pollen dispersion at time 61 for maize data
## create a ToxicIntensityRaster object
tox<-array(0,c(61,1024,1024))
tox[61,,]<-as.matrix(maize.toxicintensity61)
attr(tox,"class") <- "ToxicIntensityRaster"
## plot tox over the landscape at time 61
plot(maize.landscape,objectT=tox,time=61)

## End(Not run)
```

`plot,Landscape,Individuals-method`

plot Landscape & Individuals method

Description

Plots using Landscape & Individuals objects.

Usage

```
## S4 method for signature 'Landscape,Individuals'
plot(x, y, time = -1, objectT = NULL,
  numind = -1, add = F, ..., plot.legend = TRUE)
```

Arguments

`x`	A Landscape object
`y`	An Individuals object
`time`	Time selection (default = -1) for Toxic intensity
`objectT`	ToxicIntensityRaster, 3D array (result of toxicIntensity, default NULL)
`numind`	an individual ID
`add`	Boolean to draw hover an another plot
`...`	default plot parameters (par)
`plot.legend`	plot legend (default TRUE)

Details

If objectT and numind are informed, the function will draw an individual ecoToxical plot.

If objectT and time are specified, the function will draw the landscape, the individuals states and the toxic intensity at this time of the simulation.

By default this function will draw the landscape and the individuals positions.

Examples

```
## Not run:
data(maize_65)
# plot a landscape and individuals
plot(maize.landscape,maize.individuals)
# plot a landscape and individuals states at time 30
plot(maize.landscape,maize.individuals,time=30)
# Simulate pollen dispersion
tox <- toxicIntensity(maize.landscape,maize.emitted_pollen,1,61)
# plot a landscape, individuals and pollen dispersion of maize_65 data
plot(maize.landscape,maize.individuals,time=30,objectT=tox)
# plot ecotoxicology of individual 1.
ind2<-ecoToxic(maize.landscape,maize.individuals,objectT=tox,maxtime=61)
plot(maize.landscape,ind2,objectT=tox,numind=1)

## End(Not run)
```

`plotEcotoxic` *Plot internal toxic concentration method*

Description

Plot a time series of internal toxic concentration for a given individual.

Usage

```
plotEcoToxic(objectL, objectI, objectT, numind = 1)
```

Arguments

`objectL`	A Landscape object
`objectI`	An Individuals object
`objectT`	A ToxicIntensityRaster, a 3d array of Toxic intensity over the time [t,x,y], (first indice is time) see `toxicIntensity`
`numind`	An individual ID

Examples

```
## Not run:
data(maize_65)
# Simulate pollen dispersion
tox <- toxicIntensity(maize.landscape,maize.emitted_pollen,1,61)
# plot ecotoxicology of individual 1.
ind2<-ecoToxic(maize.landscape,maize.individuals,objectT=tox,maxtime=61)
plot(maize.landscape,ind2,objectT=tox,numind=1)

## End(Not run)
```

`Precipitation` *Precipitation data*

Description

Precipitation Data from simulation

Usage

```
data("Precipitation")
```

Format

A data frame with 4018 observations on the following 4 variables.

`AN` Year

`MOIS` Month

`JOUR` Day

`RR` Precipitation ratio

Details

Precipitation over 10 years from 2003 to 2013

Source

From simulations of the Stochastic Weather Generator developed by Allard and Bourotte (2014).

Examples

```
data(Precipitation)
```

print,Individuals-method

print Individuals informations

Description

print Individuals informations

Usage

```
## S4 method for signature 'Individuals'
print(x, ...)
```

Arguments

x	An Individuals object
...	further arguments passed to or from other methods.

Examples

```
data(maize_65)
print(maize.individuals)
```

print,Landscape-method

Print Method for Landscape

Description

Print Method for Landscape

Usage

```
## S4 method for signature 'Landscape'
print(x, ...)
```

Arguments

x	A Landscape object
...	further arguments passed to or from other methods.

Examples

```
data(maize_65)
print(maize.landscape)
```

`saveIntoFile` *Save Particles Dispersion 3D Array to tiff file*

Description

Save into tiff file particles dispersion 3D array from toxicIntensity. The output is a RasterStack with a layer per time unit with projection set to CRS="+proj=longlat +datum=WGS84"

Usage

```
saveIntoFile(objectL, objectT, filename = "ParticlesDispersion.tif",
  format = "GTiff", overwrite = T)
```

Arguments

`objectL`	a Landscape object
`objectT`	a 3D array particles dispersion indexed by time (output from toxicIntensity)
`filename`	output file name (default "ParticlesDispersion.tif")
`format`	output format (default=GTiff)
`overwrite`	if TRUE overwrite file (default TRUE)

Value

a RasterStack object

Examples

```
## Not run:
data(maize_65)
ti<-toxicIntensity(maize.landscape,maize.emitted_pollen)
saveIntoFile(maize.landscape,ti,filename="ParticlesDispersion.tiff",format="GTiff",overwrite=T)

## End(Not run)
```

`show,Individuals-method` *Show a summary of Individuals information*

Description

print a summary of Individuals information (the 10 first individuals)

Usage

```
## S4 method for signature 'Individuals'
show(object)
```

Arguments

`object`	An Individuals object

Examples

```
data(maize_65)
maize.individuals
```

`show,Landscape-method` *Show Method for Landscape*

Description

Show Method for Landscape

Usage

```
## S4 method for signature 'Landscape'
show(object)
```

Arguments

`object`	A Landscape class

Examples

```
data(maize_65)
maize.landscape
```

`simulateIndividuals` *Wrapper function SimulateIndividuals*

Description

This function simulates individuals as an Individuals object.

Will simulate `n` individuals in receptors fields of a Landscape.

Usage

```
simulateIndividuals(objectL, n = 200, mintime = 1, maxtime, dob,
  life_duration, toxic_threshold)
```

Arguments

`objectL`	A Landscape object
`n`	Number of individuals to simulate
`mintime`	Start simulation time
`maxtime`	End simulation time
`dob`	A vector for the Date Of Birth of each individual between `mintime` and `maxtime`
`life_duration`	A vector for the life duration of each individual
`toxic_threshold`	A vector for the internal toxic threshold value leading to death for each individual

Details

The Individuals object output includes for each individual the coordinates, the date of birth, the life duration, the toxic threshold and the internal toxic concentration.

Value

A S4 Individuals-class object

See Also

loadIndividuals

Examples

```
data(maize_65)
nb_ind=100 ; time_min=1 ; time_max=61
birth_dates=sample(time_min:time_max, size=nb_ind, replace=TRUE)
life_expectancies=rep(20, nb_ind)
toxic_gap=rep(15, nb_ind)
## generate exposed individuals
ind=simulateIndividuals(maize.landscape, n=nb_ind, mintime=time_min,
                        maxtime=time_max, dob=birth_dates,
                        life_duration=life_expectancies, toxic_threshold=toxic_gap)
plot(maize.landscape,ind)
```

`simulateInitialPartition` *simulateInitialPartition Method*

Description

This function creates an object Landscape and simulates a landscape with neutral and source fields.

Usage

```
simulateInitialPartition(n = 500, prop = 0.4, range = 10, xmin = 0,
  xmax = 5000, ymin = 0, ymax = 5000)
```

Arguments

`n`	Numeric, numbers of cells
`prop`	Numeric [0,1] toxic cells proportion
`range`	Aggregation parameter (range of the spatial Exponential covariance of Gaussian process)
`xmin`	x-axis left coordinates in space unit (see projections_briskaR)
`xmax`	x-axis right coordinates in space unit (see projections_briskaR)
`ymin`	y-axis bottom coordinates in space unit (see projections_briskaR)
`ymax`	y-axis top coordinates in space unit (see projections_briskaR)

Details

In the function the first step is a binomial point process to simulate a SpatialPointsDataFrame with sources and neutral marks, which depends on the aggregation parameter. The second step of the function is the Voronoi tesselation from the simulated points and returns a SpatialPolygons-DataFrame.

Value

An S4 Landscape object with `n` fields, `prop` pourcentage of toxic fields of size (xmin,xmax) (ymin,ymax)

Examples

```
## Not run:
# Simulate a 5000m x 5000m landscape with 500 cells (e.g. fields)
# whose 40% (200 cells) are sources.
# The projection by default is Lambert93 projection.
land <- simulateInitialPartition(n=500,prop=0.4,range=10,xmin=0,xmax=5000,ymin=0,ymax=5000)
plot(land)

## End(Not run)
```

`simulateLandscape` *SimulateLandscape Method*

Description

Create an object Landscape. Simulate a landscape with neutral and source fields and receptors margins.

Usage

```
simulateLandscape(n = 500, prop = 0.4, range = 10, xmin = 0,
  xmax = 5000, ymin = 0, ymax = 5000, border_size = 200,
  prob = runif(1, 0.1, 0.9), mean_thickness = runif(1, 2, 20),
  v_thickness = 50)
```

Arguments

`n`	Numeric, numbers of fields
`prop`	Numeric [0,1] toxic fields proportion
`range`	aggregation parameter (range of the spatial exponential covariance of gaussian process) in meters.
`xmin`	x-axis left coordinates
`xmax`	x-axis right coordinates
`ymin`	y-axis bottom coordinates
`ymax`	y-axis top coordinates
`border_size`	A numeric, bbox margin
`prob`	Probability to inflate a filed margin
`mean_thickness`	Margin width expectation
`v_thickness`	Margin width variance

Details

Execute both simulateInitialPartition and simulateThickMargins functions.

Value

An S4 Landscape object with n fields, prop pourcentage of toxic fields.

See Also

simulateInitialPartition and simulateThickMargins

Examples

```
## Not run:
land <- simulateLandscape(n=100, prop=0.4, range=10,
xmin=0, ymin=1000, xmax=0, ymin=1000, border_size=100,
prop=runif(1,0.1,0.9), mean_thickness=runif(1,2,20),
v_thickness=50)
plot(land)
## End(Not run)
```

`simulateThickMargins` *simulateThickMargins Method*

Description

Simulate thick margins as receptors in a landscape.

Usage

```
simulateThickMargins(objectL, ...)

## S4 method for signature 'Landscape'
simulateThickMargins(objectL, border_size = 200,
  prob = runif(1, 0.1, 0.9), mean_thickness = runif(1, 2, 20),
  v_thickness = 50)
```

Arguments

`objectL`	A Landscape object
`...`	other parameters
`border_size`	A numeric, bbox margin
`prob`	Probability to inflate a margin
`mean_thickness`	Margin width expectation in meter
`v_thickness`	Margin width variance in meter

Details

Margin width use a Gamma distribution with shape and scale parameters based on thickness mean and variance.

Value

a Landscape object with fields margin as receptor

See Also

simulateInitialPartition and simulateLandscape

Examples

```
## Not run:
data(maize_65)
plot(maize.landscape)
landscape.margin <- simulateThickMargins(maize.landscape)
plot(landscape.margin)
## End(Not run)
```

`toxicIntensity` *toxicIntensity Method*

Description

Simulate contaminants intensity over the landscape by two steps : dispersal of toxic particules and local intensity of particules after dispersal.

Usage

```
toxicIntensity(objectL, ...)

## S4 method for signature 'Landscape'
toxicIntensity(objectL, toxic_emission, mintime = 1,
  maxtime = 60, size_raster = 2^10, kernel = "NIG",
  kernel.options = list(a1 = 0.2073, a2 = 0.2073, b1 = 0.3971, b2 = 0.3971, b3
  = 0.0649, theta = 0), beta = 0.4, alpha = list(minalpha = 0.1, maxalpha =
  0.95, covariate_threshold = 30, simulate = T, covariate = NULL))
```

Arguments

`objectL`	A Landscape object
`...`	parameters
`toxic_emission`	Matrix of sources emissions, row as sources ID, col as time
`mintime`	Start simulation time (default=1)
`maxtime`	End simulation time
`size_raster`	raster size (default = 2^10)
`kernel`	dispersion kernel, function name (default = NIG)
`kernel.options`	parameters list for the kernel function
`beta`	toxic adherence parameter between 0 and 1 (default = 0.4)
`alpha`	list of toxic loss options (default = list(minalpha=0.1,maxalpha=0.95,covariate_threshold=30,simulate=TRUE,covariate=NULL))

Details

The dispersal of contaminants is implemented by rastering the landscape and by computing the convolution between sources emissions and a dispersal kernel.

The dispersion kernel by default is Normal Inverse Gaussian kernel ("NIG" function). Currently, two others are implemented "geometric" (with parameter `a`) and "2Dt" kernels (with parameters `a`, `b`, `c1`, `c2`).

Local intensity depends of `beta` and `alpha` parameters. Beta represents the toxic adherence between [0,1]. Alpha represents a list of parameters of the lost of toxic particules due to covariates (precipitation). There are two configurations to integrate the loss in the function : (i) simulating covariate (simulate=TRUE) or (ii) uploading covariate (simulate=FALSE). The covariate is linked to the loss by a linear regression with paramaters minalpha, maxalpha, covariate_threshold.

Value

A ToxicIntensityRaster, a 3D array as time matrix dispersion, [t,x,y]

Examples

```
## Not run:
data(maize_65)
data(maize.emitted_pollen)
raster.size <- 1024
tox <- toxicIntensity(maize.landscape,maize.emitted_pollen,
mintime=1,maxtime=61,size_raster=raster.size)
# plot particles dispersion at time 30
image(x=1:raster.size, y=1:raster.size, z=tox[30,,])
# plot the landscape and the pollen dispersion at time 61
plot(maize.landscape,objectT=tox,time=61)

## End(Not run)
```

`[,Individuals` *Get an individual information*

Description

Get an Individual information

x[i] get individual i values from Individuals x.

Get an Individual information at a specified time

x[i,j] get individual i at time j values from Individuals x.

Usage

```
## S4 method for signature 'Individuals,numeric,missing,ANY'
x[i]

## S4 method for signature 'Individuals,numeric,numeric,ANY'
x[i, j]
```

Arguments

x	An Individuals object
i	individual index
j	time of information

Value

a data.frame with all Individuals slot(attributes) for an individual

a data.frame with all Individuals slot (attributes) for an individual at a specified time.

Examples

```
data(maize_65)
# get individual 99 informations
str(maize.individuals[99])
data(maize_65)
#get individual 99 informations at time 30
maize.individuals[99,30]
```

Index