Package 'mateable'

February 4, 2023

Type Package

Title Assess Mating Potential in Space and Time

Version 0.3.2

Date 2023-01-04

Maintainer Stuart Wagenius <stuart.wagenius@gmail.com>

URL https://github.com/stuartWagenius/mateable

BugReports https://github.com/stuartWagenius/mateable/issues

Description Simulate, manage, visualize, and analyze spatially and temporally explicit datasets of mating potential. Implements methods to calculate synchrony, proximity, and compatibility.Synchrony calculations are based on methods described in Augspurger (1983) <doi:10.2307/2387650>, Kempenaers (1993) <doi:10.2307/3676415>, Ison et al. (2014) <doi:10.3732/ajb.1300065>, and variations on these, as described.

License GPL

LazyData true

Imports FNN, graphics, grDevices, Rcpp, sn, stats

Depends R(>= 2.10.0)

Suggests knitr, rmarkdown

LinkingTo Rcpp

RoxygenNote 7.2.3

VignetteBuilder knitr

Encoding UTF-8

Biarch true

NeedsCompilation yes

Author Stuart Wagenius [cre, aut],
Danny Hanson [aut],
Amy Waananen [aut]

Repository CRAN

Date/Publication 2023-02-04 07:42:31 UTC

R topics documented:

`mateable-package` *Tools to simulate, manage, visualize, and jointly analyze spatially and temporally explicit datasets of mating potential*

Description

This package facilitates the investigation of three dimensions of mating potential. It provides a method for simulating populations and includes a dataset.

Author(s)

Stuart Wagenius, Danny Hanson, Amy Waananen

References

Background: https://echinaceaProject.org/

Examples

```
library(mateable)

pop <- simulateScene()
pop

plotScene(pop)
plotScene(pop, "t")
```

```
plot3DScene(pop)

sync <- synchrony(pop, "augs")
prox <- proximity(pop, "maxPropSqrd")
plotPotential(sync)
plotPotential(prox, "ind")


str(ech2012)
ee <- makeScene(ech2012, FALSE, "firstDay", "lastDay", "Ecoord", "Ncoord", idCol = "tagNo")
```

`compatibility` *Make potentials object–mating type compatibility*

Description

Calculate one of several measures of mating compatibility.

Usage

```
compatibility(scene, method, subject = "all", averageType = "mean")
```

Arguments

`scene`	a matingScene object
`method`	either "si_echinacea" or "dioecious" see details for further description
`subject`	whether you want pair, individual, population, or all. Specifying more than one is allowed.
`averageType`	whether to calculate individual and population proximity using the mean or median

Details

When `method` is "si_echinacea" compatibility will be calculated as sporophytic self incompatible (si) in the same manner as Echinacea (and many other plants). For two individuals, they are incompatible if they share any S alleles (columns s1 and s2) and they compatible otherwise. When `method` is "dioecious" it is assumed that the column s1 will contain either a 1 or 2 depending on the individual's sex. Thus, when comparing two individuals, they are compatible if s1 of the first != s1 of the second, and s2 is ignored.

Value

A potentials object containing one more more of the following, depending the input for `subject`: If `subject` is "population" the return list will contain a numeric value that has a range depending on the `method`. If `subject` is "pair" the return list will contain a matrix with all pairwise compatibilities. If `subject` is "individual" the return list will contain a dataframe with a column containing IDs and a column containing compatibility averages. If `subject` is "all" the return list will contain all three of the items above.

Author(s)

Danny Hanson

Examples

```
pop <- simulateScene()
compatibility(pop, "si_echinacea")
```

`dailyOMP` *Calculate daily outcrossed mating potential*

Description

dailyOMP generates an OMP object giving the daily outcrossed mating potential of individuals based on k nearest neighbors

Usage

```
dailyOMP(
  scene,
  k = 3,
  days = NULL,
  gamma = 1/13.3,
  nn.constant = FALSE,
  sum = FALSE,
  mean = FALSE
)
```

Arguments

`scene`	a matingScene object
`k`	integer, number of nearest neighbors to use in calculating OMP
`days`	the day or range of days to calculate OMP for, default is all days in a scene (see Details)
`gamma`	parameter of exponential decay to be used in calculating OMP (defaults to 1/13)
`nn.constant`	logical; indicates whether the nearest neighbors used in calculations should be the nearest on a given day (`nn.constant = FALSE`) or the nearest neighbors over an entire season (`nn.constant = TRUE`)
`sum`	logical; indicates if the return should be a sum of an individual's daily outcrossed mating potential over the range specified by days
`mean`	logical; indicates if the return should be the mean of an individual's daily outcrossed mating potential over the range of days that the individual was receptive to mating

Details

Daily outcrossed mating potential is a weighted average of an individual's distance to their nearest neighbors on a given day (Wagenius et al. 2007). The days to calculate OMP for should be input as integers relative to the first day of flowering, as they are in the start and end columns of a matingScene object. If the number of ids receptive on a day is less than k, OMP will be calculated for the maximum number of neighbors.

Value

a named matrix with a row for each id and a column for each day, and entries corresponding to ids' OMP each day

Author(s)

Amy Waananen

References

Wagenius, S., E. Lonsdorf, and C. Neuhauser. 2007. Patch aging and the S-Allee effect: breeding system effects on the demographic response of plants to habitat fragmentation. *American Naturalist* **169**:383-397.

See Also

`makeScene`, `proximity`, `synchrony`, `receptivityByDay`

Examples

```
pop <- simulateScene()
omp <- dailyOMP(pop)
omp.1 <- dailyOMP(pop, nn.constant = TRUE) # same nearest neighbors throughout the season
omp.2 <- dailyOMP(pop, nn.constant = FALSE) # nearest flowering neighbors
```

---

ech2012 *Information about mating scene at sites eelr and nwlf in 2012.*

Description

This dataframe contains information about all 53 plants that flowered in 2012 at the sites eelr (East Elk Lake Road) and Northwest of Landfill (nwlf). Kelly Kapsar visited plants regularly to determine the starting and ending dates of flowering on every head of every plant. The metadata for the phenology dataset can be provide upon request to the maintainer. Plants were mapped with gps with better than 6 cm precision.

Usage

```
ech2012
```

Format

A 53 x 6 data frame

Variables

Variables:

- tag, unique identifier for each plant
- pop, population corresponding to site
- firstDay, the first day that any head on the plant shed pollen
- lastDay, the last day that any head on the plant shed pollen
- Ecoord, the x-coordinate of each plant in meters
- Ncoord, the y-coordinate of each plant in meters

References

Wagenius, S. 2006. Scale-dependence of reproductive failure in fragmented *Echinacea* populations. Ecology 87: 931-941.

Examples

```
dim(ech2012)
str(ech2012)
```

---

kNearNeighbors *Get k Nearest Neighbors*

Description

Find the k nearest neighbors for all individuals in a population. This function is simply a wrapper for `FNN::knn.dist`.

Usage

```
kNearNeighbors(scene, k)
```

Arguments

`scene`	a matingScene object
`k`	integer of how many nearest neighbors to get

Value

a matrix where the rows are all individuals and the columns are their k nearest neighbors

See Also

knn.dist, proximity

Examples

```
pop <- simulateScene(10)
kNearNeighbors(pop, 3)
```

---

makeScene *Create a matingScene object from a data frame*

Description

Turns a data frame with information about temporal, spatial, or genetic mating data into a matingScene object using a standard format.

Usage

```
makeScene(
  df,
  multiYear = FALSE,
  startCol = "start",
  endCol = "end",
  xCol = "x",
  yCol = "y",
  s1Col = "s1",
  s2Col = "s2",
  idCol = "id",
  otherCols = NULL,
  dateFormat = "%Y-%m-%d",
  split = NULL
)
```

Arguments

df	a data frame containing information about a mating scene, namely coordinate of individuals in space, time, and mating type.
multiYear	logical indicating whether or not to split the result into a list by year
startCol	character name of column with start dates
endCol	character name of column with end dates
xCol	character name of column with x or E coordinates
yCol	character name of column with y or N coordinates
s1Col	character name of one column with S-allele
s2Col	character name of another column with S-alleles

idCol	character name for column with unique identifier
otherCols	character vector of column(s) to include besides the necessary ones for the mating scene. If NULL, it will be ignored.
dateFormat	character indicating either (1) the format of the start and end date columns if those columns are characters or (2) the origin for the start and end date columns if those columns are numeric. It is used in as.Date
split	character name for a column with values by which the result should be split

Details

The input dataframe can contain information about locations of individuals in 1, 2, or 3 dimensions of a mating scenes. The function currently allows two spatial coordinates. The user specifies the names of the columns and they will be saved xCol and yCol in the matingScene object. MatingScene objects currently save temporal coordinates for individuals as start and end date of mating activity within a year. Mating type coordinates are saved as mating type alleles. Columns are named id, start, end, x, y, s1, and s2 for idCol, startCol, endCol, xCol, yCol, s1Col, and s2Col respectively. The attributes "t", "s", and "mt" will be set to TRUE if the data frame has temporal, spatial, or mating type data, respectively and will be FALSE otherwise. The attribute originalNames contains all the names of the columns in the original data frame.
The start and end columns will be changed to integers relative to the start day of the population. So the first day of the first individual to become receptive will be 1 and so on. The attribute origin contains the origin that can be used when converting the columns start and end from integers to dates.
If no temporal data are available except the year in which it was collected and df is a multi-year data set, put the collection year into the column labeled as startCol and set dateFormat = " the data appropriately.

Value

a matingScene object, either a single dataframe in standard format or a list of dataframes. Attributes of the matingScene object indicate the type of information in the data frame, including the original column names, and the origin of the date columns. If multiYear = TRUE, the return value will be a list of matingScene data frames where each element in the list represents one year. If split is specified, the return value will be a list of matingScene data frames where each element in the list represents a value of the specified variable. See details for more information on attributes and how to work with multi-year data.

Author(s)

Danny Hanson

---

matingSummary *Summarize a Mating Scene*

Description

Create a summary of information contained within a matingScene object.

Usage

```
matingSummary(
  scene,
  type = "auto",
  k = 1,
  compatMethod = "si_echinacea",
  as.data.frame = FALSE
)
```

Arguments

scene	a matingScene object
type	character. whether to do a temporal (t), spatial (s), or mating type (mt) summary. The default is "auto" which will automatically summarize all mating information in scene
k	integer. Which nearest neighbor to calculate (only for type == "s")
compatMethod	character indicating the method to use when calculating compatiblity. Defaults to "si_echinacea"
as.data.frame	logical. If TRUE, returns summary as a dataframe.

Value

a list, list of lists, or dataframe containing summary information including:
temporal - year (year), population start date (popSt), mean individual start date (meanSD), standard deviation of start (sdSD), mean duration (meanDur), standard deviation of duration (sdDur), peakDay - day(s) on which highest number of individuals were receptive (peak), mean end date (meanED), standard deviation of end date (sdED), population end date (popEnd)
spatial - minimum x (minX), minimum y (minY), maximum x (maxX), maximum y (maxY), average distance to kth nearest neighbor as specified by k (k<n> where n is the input for k)
compatibility - number of mating types (nMatType), average number of compatible mates (meanComp)
If scene is a multi-year matingScene, then the output will be a list of lists, one list for each year. If as.data.frame = TRUE, the output will be a dataframe with columns containing summary information and, if applicable, an 'id' column identifying what portion of the matingSummary object it summarized. If the scene is a multi-year matingScene, then the output will be a list of dataframes, one list for each year.

Examples

```
eelr <- makeScene(ech2012, startCol = "firstDay", endCol = "lastDay",
  xCol = "Ecoord", yCol = "Ncoord", idCol = "tagNo")
eelrSum <- matingSummary(eelr)
eelrSum[c("minX", "minY", "maxX", "maxY")]
```

`overlap` *Pairwise Mating Timing Comparison*

Description

Get comparisons of mating timing between all pairs

Usage

```
overlap(scene, overlapOrTotal = c("overlap", "total"), compareToSelf = FALSE)
```

Arguments

`scene`	a matingScene object
`overlapOrTotal`	whether to calculate the number of days that each pair was overlapping in mating receptivity or the total number of days that either individual was receptive
`compareToSelf`	whether or not to include self comparisons in the return value

Value

a matrix containing all pairwise comparisons. If compareToSelf is FALSE then there will be n rows and n-1 columns.
To index result[i,j] where j > i, use `result[i,j-1]`, where `result` is the return value of overlap.
There is one attribute "idOrder" which holds the order of the id column in scene at the time of the function call. This can be useful to find certain elements in the matrix (see examples).
If scene is a multi-year matingScene, then overlap will return a list of matrices (as described above) where each matrix represents one year.

Author(s)

Danny Hanson

Examples

```
pop <- simulateScene()
pop <- pop[order(pop$start),]
daysSync <- overlap(pop)
indices <- which(attr(daysSync, "idOrder") %in% c(1, 4))
if (indices[1] <= indices[2]) {
  daysSync[indices[1], indices[2]]
} else {
  daysSync[indices[1], indices[2]-1]
}
```

pairDist *Distance Matrix for a mating scene*

Description

Compute all pairwise distances for a population. This function is simply a wrapper for `dist` that returns only a matrix

Usage

```
pairDist(scene)
```

Arguments

`scene`	a matingScene object

Value

a matrix of all pairwise comparisons with attributes for order of identifiers (idOrder)

See Also

`dist`

Examples

```
pop <- simulateScene()
distance <- pairDist(pop)
```

plot3DPotential *graphical visualization of multiple mating potential objects*

Description

Visualize multiple dimensions of mating potential

Usage

```
plot3DPotential(
  matPots,
  subject = NULL,
  density = TRUE,
  sub.ids = NULL,
  N = 3,
  sample = NA,
  text.cex = 0.7,
  pt.cex = 0.7
)
```

Arguments

matPots	list, contains one or multiple mating potential objects representing unique potential dimensions
subject	character, indicates whether the subject to be visualized is individuals (subject = 'ind') or all pairwise interactions (subject = 'pair')
density	logical, if TRUE (default), plots probability density over histogram
sub.ids	vector, contains the IDs of individuals to be represented in pairwise potential plots
N	integer, indicates the number of individuals to sample if sub.ids = 'random' (default N = 3)
sample	character, specifies how to sample individuals to be represented in pairwise potential plots. Possible values are "random" (default) or "all". See details.
text.cex	specify text expansion factor (text size relative to device default)
pt.cex	specify point expansion factor (point size relative to device default)

Details

The individuals to be represented in the pairwise potential plots can either be specified explicitly through sub.ids, chosen randomly (sample = 'random'), or all individuals can be selected (sample = 'all'). The default is to randomly select 9 individuals. If multiple years are being plotted, the subset is sampled from all years and the same individuals will be represented in each year, if possible.

Value

No return value, called to draw a plot

Author(s)

Amy Waananen

See Also

see generic function points for values of pch

Examples

```
pop <- simulateScene()
sync <- synchrony(pop, "augs")
prox <- proximity(pop, 'maxProp')
compat <- compatibility(pop, 'si_echinacea')
plot3DPotential(list(sync,prox,compat), subject = 'ind')
```

plot3DScene *multi-dimensional visualization of mating scene object*

Description

Visualize multiple dimensions of a mating scene

Usage

```
plot3DScene(
  scene,
  dimension = "auto",
  sub = NULL,
  N = 3,
  ycoord = "northing",
  xcoord = "easting",
  pch = 19,
  pt.cex = 0.7,
  label.cex = 0.7,
  mt1 = "F",
  mt2 = "M",
  plot.lim.zoom = FALSE,
  ...
)
```

Arguments

scene	a matingScene object
dimension	what dimension(s) of the mating scene should be visualized. Possible dimensions are 't' for temporal, 's' for spatial, 'mt' for mating type, and 'auto' (the default). For dimension = 'auto', all dimensions represented in the mating scene object will be plotted.
sub	a subset of the population to plot; either a character indicating whether to subset a random sample (sub='random'), all individuals (sub='all'), or a vector containing the IDs of the individuals to subset.
N	if sub = 'random', the number of individuals to sample (default N = 3)
ycoord	y-axis coordinate system label
xcoord	x-axis coordinate system label
pch	point type, defaults to pch = 19, solid filled in circle. If pch = NULL, individuals will be labeled by their id.
pt.cex	specify point expansion factor (point size relative to device default)
label.cex	specify text expansion factor (text size relative to device default)
mt1	label for mating type '1', if dioecious; defaults to 'F'
mt2	label for mating type '2', if dioecious; defaults to 'M'

plot.lim.zoom	if TRUE, spatial plot limits for lists of scenes are set by the maximum from all scenes
...	optional arguments for the plot function

Value

No return value, called to draw a plot

Author(s)

Amy Waananen

See Also

see generic function points for values of pch

Examples

```
pop <- simulateScene()
plot3DScene(pop)
```

---

plotPotential *graphical visualization of a mating potential object*

Description

Visualize mating potential

Usage

```
plotPotential(
  matPot,
  subject = NULL,
  plotType = "auto",
  density = TRUE,
  sub.ids = NULL,
  N = 9,
  sample = "random",
  ...
)
```

Arguments

matPot	a mating potential object
subject	character, either 'ind' or 'pair', indicating whether the subject being visualized is individuals or pairwise interactions
plotType	character, indicating what plots are to be displayed. See details. Options are histogram ('hist'), network diagram ('net'), and heatmap ('heat'). If mating potential object
density	logical. If TRUE (default), plots probability density over histogram.
sub.ids	a vector containing the ids of individuals to be represented in pairwise potential plots
N	a positive number indicating the number of individuals to sample if sub.ids = 'random'
sample	a character string specifying how to choose a subset of individuals to be represented in pairwise potential plots. Possible values are "random" (default) or "all" (see details).
...	optional arguments for the plot function

Details

Options for plotType are 'hist' (histogram), 'net' (network diagram), 'heat' (heatmap), and 'auto'. Default value is 'auto': if the mating potential object contains pairwise potential, 'auto' returns all plot types, otherwise it returns histograms of individual potential.

The individuals to be represented in the pairwise potential plots can either be specified explicitly through sub.ids, chosen randomly (sample = 'random'), or all individuals can be selected (sample = 'all'). The default is to randomly select 9 individuals. If multiple years are being plotted, the subset is sampled from all years and the same individuals will be represented in each year, if possible. If fewer than three individuals from the subset are available in a year, no network diagram or heatmap will be returned for that year.

Value

No return value, called to draw plots

Author(s)

Amy Waananen

See Also

see generic function points for values of pch

Examples

```
pop <- simulateScene()
sync <- synchrony(pop, "augs")
plotPotential(sync)
```

plotScene *graphical visualization of a mating scene object*

Description

Visualize a mating scene

Usage

```
plotScene(
  scene,
  dimension = "auto",
  dailyPoints = TRUE,
  drawQuartiles = TRUE,
  sortBy = c("start", "end"),
  colorBy = NULL,
  sub = NULL,
  N = 3,
  label.sub = TRUE,
  xlab.spat = NULL,
  ylab.spat = NULL,
  pch = 19,
  pt.cex = 0.75,
  label.cex = 0.8,
  plot.lim.zoom = FALSE,
  quartile.lwd = 1,
  quartile.col = "gray55",
  peak.col = "gray27",
  labelID = FALSE,
  mt1 = "F",
  mt2 = "M",
  leg.ncol = 1,
  ...
)
```

Arguments

scene	a matingScene object
dimension	what dimension(s) of the mating scene should be visualized. Possible dimensions are 't' for temporal, 's' for spatial, 'mt' for mating type, and 'auto' (the default). For dimension = 'auto', all dimensions represented in the mating scene object will be plotted.
dailyPoints	logical indicating whether daily counts of individuals should be displayed for plots of the temporal dimension
drawQuartiles	logical indicating whether vertical lines should be drawn at population peak (see details) or quartiles

sortBy	character indicating which columns to sort segments of flowering schedule by. Defaults to 'start', then 'end'. Up to three variables may be specified.
colorBy	character optional, the name of a variable to use to assign color to segments or points.
sub	a vector containing the ids of individuals to be highlighted in the plots or a character string specifying how to choose individuals to highlight. Possible values are "random" or "all". If NULL, no subset will be identified in the plots.
N	a positive number, the number of individuals to sample if sub = 'random'
label.sub	logical, indicating whether specified subset should be labeled
xlab.spat	character label for x-axis of spatial dimension plots. If NULL, defaults to 'easting'.
ylab.spat	character label for y-axis of spatial dimension plots. If NULL, defaults to 'northing'.
pch	specify point type to be used in plots. Defaults to pch = 19 (filled-in circle). If NULL, points will be labeled with their id.
pt.cex	specify point expansion factor (point size relative to device default)
label.cex	specify text expansion factor (text size relative to device default)
plot.lim.zoom	if TRUE, spatial plot limits for lists of scenes are set by the maximum from all scenes
quartile.lwd	if drawQuartiles = TRUE, specifies weight of quartile and peak lines relative to device default.
quartile.col	if drawQuartiles = TRUE, specifies color of quartile lines, defaults to 'gray81'.
peak.col	if drawQuartiles = TRUE, specify color of peak lines, defaults to 'gray27'.
labelID	if TRUE, the y-axis will be labeled with the id of the corresponding segment.
mt1	label for mating type '1', if dioecious
mt2	label for mating type '2', if dioecious
leg.ncol	number of columns to include in legend, if colorBy is not NULL
...	standard graphical parameters

Details

Population peak is defined by when maximum number individuals were reproductively receptive on one day. If multiple days had the same maximum number, peak is defined as the median of these dates.

Value

No return value, called to draw a plot

Author(s)

Amy Waananen

See Also

see plot3DScene to visualize multiple dimensions on one plot

Examples

```
pop <- simulateScene()
plotScene(pop)
```

`proximity` *Make potentials object–spatial proximity*

Description

Calculate one of several measures of spatial proximity

Usage

```
proximity(
  scene,
  method,
  averageType = "mean",
  subject = "all",
  zeroPotDist = NULL,
  k = 6
)
```

Arguments

`scene`	a matingScene object
`method`	one of "maxProp", "maxPropSqrd", or 'knn.dist'; see details for further description
`averageType`	whether to calculate individual and population proximity using the mean or median
`subject`	whether you want pair, individual, population, or all. Specifying more than one is allowed.
`zeroPotDist`	the distance at which potential should be equal to zero
`k`	the number of the nearest neighbor to search, if `method` is "knn.dist". Defaults to 6, but must be less than population size.

Details

If `method` is "maxProp" then proximity between two individuals will be calculated as 1 - distance/max(distance). If `method` is "maxPropSqrd" then proximity between two individuals will be calculated as (1 - distance/max(distance))^2. If `method` is "knn.dist" then the function This uses `FNN::knn.dist` to return the Euclidian distance of the kth nearest neighbor.

Value

A potentials object containing one more more of the following, depending the input for `subject`: If `subject` is "population" the return list will contain a numeric value that has a range depending on the `method`. If `subject` is "pair" the return list will contain a matrix with all pairwise proximity comparisons. If `subject` is "individual" the return list will contain a dataframe with a column containing IDs and a column containing proximity values. If `subject` is "all" the return list will contain all three of the items above.

Author(s)

Danny Hanson

Examples

```
pop <- simulateScene()
proximity(pop, "maxProp")
```

`receptivityByDay` *Mating Receptivity by Day*

Description

Create a matrix showing which individuals are receptive on a given day.

Usage

```
receptivityByDay(scene, summary = FALSE, nameDate = TRUE)
```

Arguments

`scene`	a matingScene object
`summary`	logical, summarizes number of receptive individuals on each day
`nameDate`	logical, if summary = TRUE, option to name indices of the vector by the date they represent (rather than named relative to first day of receptivity in a season)

Value

if summary = FALSE (default), a matrix where the columns represent all mating days and the rows represent all individuals in the population. If summary = TRUE, a named vector where each index gives the number of receptive individuals on a given day and is named by the day it represents. If a matrix, the value at position [i,j] will be TRUE if individual j was receptive on day i
If scene is a multi-year matingScene, then receptivityByDay will return a list of matrices (as described above) where each matrix represents one year.

Author(s)

Danny Hanson, Amy Waananen

Examples

```
pop <- simulateScene(size = 10)
receptivityByDay(pop)
```

`simulateScene` *Simulate a Mating Scene*

Description

simulateScene generates a matingScene object – a simulated population in a standard format with individuals randomly assigned a mating schedule, a location, and S-alleles

Usage

```
simulateScene(
  size = 30,
  meanSD = "2012-07-12",
  sdSD = 6,
  meanDur = 11,
  sdDur = 3,
  skSD = 0,
  xRange = c(0, 100),
  yRange = c(0, 100),
  distro = "unif",
  sAlleles = 10
)
```

Arguments

`size`	integer number of plants
`meanSD`	date mean start date
`sdSD`	date standard deviation of start date
`meanDur`	numeric duration in days
`sdDur`	standard deviation of duration in days
`skSD`	skew of the start date of the population
`xRange`	range of spatial extent of individuals along x-axis
`yRange`	range of spatial extent of individuals along y-axis
`distro`	unimplemented
`sAlleles`	integer count of S-Alleles that could be in the population

Value

matingScene data frame – see `makeScene`

Author(s)

Stuart Wagenius

See Also

makeScene

Examples

```
simulateScene()
```

---

synchrony *Make potentials object–mating synchrony*

Description

Calculate one of a variety of measures of mating synchrony.

Usage

```
synchrony(
  scene,
  method,
  subject = "all",
  averageType = "mean",
  syncNN = 1,
  compareToSelf = FALSE,
  frame = "within",
  resolution = "daily"
)
```

Arguments

scene	a matingScene object that includes the flowering schedule for the scene of interest.
method	character, partial matching allowed, describing what type of synchrony will be calculated. "augspurger" is based on the method described in Augspurger (1983). "kempenaers" is based on the method described in Kempenaers (1993). "sync_prop" will calculate individual synchrony based on the proportion of the sum of all individuals' days available to mate that coincided with the individual's days available for mating. "overlap" is based on the method described in Ison et al. (2014) and will calculate a synchrony value based on the number of days both individuals were flowering divided by the number of days either individual was available for mating. "sync_nn" gives the average of the kth nearest neighbor, or rather the kth most synchronous individual. "peak-n" will calculate the number of individuals receptive on the peak day (day with highest mating receptivity) divided by the number of individuals in the population. "peak-observations" will

	calculate the number of individuals receptive on the peak day divided by the total number of observations - this method is useful for comparing to data that has no information on individuals. "average-peak" calculates the average (determined by argument `averageType`) number of individuals receptive per day divided by the maximum number of individuals receptive per day. All "simple" methods do not have pairwise or individual values. "mean_interactions" gives the mean number of potential mating interactions an individual obtains per unit time for the period that the individual was flowering.
`subject`	one of "population", "pairwise", "individual", or "all" - see Value for more details.
`averageType`	character. Identifies whether to take the mean or median when calculating averages
`syncNN`	integer between 1 and n-1 (inclusive) or numeric between 0 and 1 (exclusive). The kth nearest neighbor to be averaged when calculating population synchrony. If k is in (0,1) then the k*nth nearest neighbor will be found
`compareToSelf`	logical. Whether or not to include self comparisons when calculation synchrony. Defaults to FALSE.
`frame`	the timeframe that synchrony is to be calculated over; options are 'within,' for synchrony within a season, or 'between,' for synchrony across multiple seasons. Defaults to 'within'.
`resolution`	if `method = sync_prop`, indicates whether temporal resolution should be yearly or daily

Details

Measures of synchrony are based on methods described in Augspurger (1983), Kempenaers (1983), and from Ison et al. (2014), as well as variations on different factors of those measures.

Value

A potentials object containing one more more of the following, depending the input for `subject`: If `subject` is "population" `synchrony` will return a numeric value that has a range depending on the `method`. If `subject` is "pairwise" `synchrony` will return a matrix with all pairwise synchrony comparisons. It is important to note two things: [1] if `method` is set to "sync_nn" then the pairwise comparisons will be in descending order and cannot be indexed by ID order. [2] if `compareToSelf` is set to FALSE, the matrix will have dimensions 100 rows by 99 columns. Similar to `overlap`, indexing will be affected. If `subject` is "individual" `synchrony` will returns a data frame with a row for id and a row for individual synchrony. If `subject` is "all" `synchrony` will return a list containing the values described above for population, pairwise, and individual synchrony.

Author(s)

Danny Hanson, Amy Waananen

References

Augspurger, C.K. (1983) Phenology, flowering synchrony, and fruit set of six neotropical shrubs. *Biotropica* **15**, 257-267.

Ison, J.L., S. Wagenius, D. Reitz., M.V. Ashley. (2014) Mating between *Echinacea angustifolia* (Asteraceae) individuals increases with their flowering synchrony and spatial proximity. *American Journal of Botany* **101**, 180-189

Kempenaers, B. (1993) The use of a breeding synchrony index. *Ornis Scandinavica*, **24**, 1.

Examples

```
pop <- simulateScene(size = 150)
synchrony(pop, "augs")

pop2 <- simulateScene(size = 1234, sdDur = 5, sk = 1)
syncVals <- synchrony(pop2, "sync_nn", "all", "median", 123)
```

Index