Package 'swaRm'

January 11, 2023

Type Package

Title Processing Collective Movement Data

Version 0.6.0

Date 2023-01-10

Maintainer Simon Garnier <garnier@njit.edu>

Description Function library for processing collective movement data (e.g. fish schools, ungulate herds, baboon troops) collected from GPS trackers or computer vision tracking software.

License GPL-3

Imports stats, splancs, geosphere, lubridate, MASS

Suggests covr, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

URL https://swarm-lab.github.io/swaRm/, https://github.com/swarm-lab/swaRm

BugReports https://github.com/swarm-lab/swaRm/issues

RoxygenNote 7.2.3

Encoding UTF-8

Config/testthat/edition 3

NeedsCompilation no

Author Simon Garnier [aut, cre] (<https://orcid.org/0000-0002-3886-3974>)

Repository CRAN

Date/Publication 2023-01-11 10:20:04 UTC

R topics documented:

swaRm-package 2
.cartesianPerimeter 3
.cartesian_perimeter 4
.ellipse 4

ang_acc 5
ang_speed 6
centroid 7
chull_area 8
chull_perimeter 9
dist2centroid 10
heading 11
is_chull 12
linear_acc 13
linear_dist 14
linear_speed 15
nn 16
nnd 17
nsd 18
pdist 19
pol_order 20
sphericity 20
stretch 21

Index **23**

`swaRm-package` *Analysis tools for collective animal movement data*

Description

This package contains functions to facilitate and automate the analysis of collective animal movement data.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

Useful links:

- https://swarm-lab.github.io/swaRm/
- https://github.com/swarm-lab/swaRm
- Report bugs at https://github.com/swarm-lab/swaRm/issues

`.cartesianPerimeter` *Rotational Order Parameter*

Description

Given a set of headings and locations, this function returns the rotational order of the set

Usage

```
.cartesianPerimeter(x, y)

rot_order(x, y, h)

rotOrder(h, x, y)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
h	A vector of headings (in radians).

Value

A single value between 0 and 1 corresponding to the rotational order parameter of the group.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

pol_order

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
h <- runif(25, 0, 2 * pi)
rot_order(x, y, h)
```

.cartesian_perimeter *Perimeter Of A Polygon In Cartesian Space*

Description

Given a set of Cartesian coordinates representing a polygon, this function computes the perimeter of the polygon.

Usage

```
.cartesian_perimeter(x, y)
```

Arguments

x	A vector of x coordinates.
y	A vector of y coordinates.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

chull_perimeter

.ellipse *Bivariate Confidence Ellipse*

Description

This function computes the confidence ellipse of a set of bivariate coordinates.

Usage

```
.ellipse(x, y, level = 0.95)
```

Arguments

x	A vector of x coordinates.
y	A vector of y coordinates.
level	The confidence level of the ellipse.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

sphericity, stretch

---

ang_acc *Angular Acceleration*

Description

Given a set of locations defining a trajectory, this function approximates their instantaneous angular accelerations computed as the difference between successive angular speeds.

Usage

```
ang_acc(x, y, t, geo = FALSE)

angAcc(x, y, t, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
t	A vector of timestamps corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x, y and t corresponding to the approximated instantaneous angular accelerations along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

heading, ang_speed

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
t <- as.POSIXct(1:25, origin = Sys.time())
ang_acc(x, y, t)
```

ang_speed *Angular Speeds*

Description

Given a set of locations defining a trajectory, this function approximates their instantaneous instantaneous angular speeds computed as the difference between successive headings divided by the time between these successive headings.

Usage

```
ang_speed(x, y, t, geo = FALSE)

angSpeed(x, y, t, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
t	A vector of timestamps corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x, y and t corresponding to the approximated instantaneous angular speeds along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

heading, ang_acc

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
t <- as.POSIXct(1:25, origin = Sys.time())
ang_speed(x, y, t)
```

centroid *Centroid*

Description

This function computes the centroid (or center of mass) of a set of x-y (or longitude-latitude) coordinates.

Usage

```
centroid(x, y, robust = FALSE, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
robust	A logical value indicating whether to compute the centroid as a simple average of the coordinates (FALSE, the default), or as the average of the coordinates weighted by the inverse of their mean pairwise distance to all other coordinates in the set (TRUE).
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A two-element list corresponding to the location of the centroid.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

dist2centroid

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
centroid(x, y)
```

chull_area *Surface Area of the Convex Hull*

Description

Given a set of locations, this function determines the surface area of the convex hull (or envelope) of the set.

Usage

```
chull_area(x, y, geo = FALSE)

chullArea(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). If TRUE, the surface area is returned as square meters. If FALSE, it is returned as square units of the [x,y] coordinates. Default: FALSE.

Value

A single numeric value corresponding to the surface area of the convex hull (in square meters if geo is TRUE).

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

is_chull, chull_perimeter

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
chull_area(x, y)
```

chull_perimeter *Perimeter of the Convex Hull*

Description

Given a set of locations, this function determines the perimeter of the convex hull (or envelope) of the set.

Usage

```
chull_perimeter(x, y, geo = FALSE)

chullPerimeter(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). If TRUE, the perimeter is returned as meters. If FALSE, it is returned as units of the [x,y] coordinates. Default: FALSE.

Value

A single numeric value corresponding to the perimeter of the convex hull (in meters if geo is TRUE).

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

is_chull, chull_area

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
chull_perimeter(x, y)
```

dist2centroid *Distances to the Centroid*

Description

Given a set of x-y (or longitude-latitude) coordinates, this function computes their respective distance to the centroid (or center of mass) of the set.

Usage

```
dist2centroid(x, y, robust = FALSE, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
robust	A logical value indicating whether to compute the centroid as a simple average of the coordinates (FALSE, the default), or as the average of the coordinates weighted by the inverse of their mean pairwise distance to all other coordinates in the set (TRUE).
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the individual distance of each point to the centroid of the set.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

centroid

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
dist2centroid(x, y)
```

heading *Headings*

Description

Given a set of locations defining a trajectory, this function approximates their instantaneous headings computed as the direction of the vectors between successive locations along the trajectory.

Usage

```
heading(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the approximated headings along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

ang_speed, ang_acc

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
heading(x, y)
```

is_chull *Points on Convex Hull*

Description

Given a set of locations, this function determines which locations belongs to the convex hull (or envelope) of the set.

Usage

```
is_chull(x, y)

isChull(x, y)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.

Value

A numerical vector of the same length as `x` and `y`. `0` indicates that the corresponding location is not part of the convex hull of the set. Values `>0` indicates that the corresponding location is part of the convex hull, and each value corresponds to the order of the locations along the convex hull polygon.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

chull_area, chull_perimeter

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
is_chull(x, y)
```

linear_acc *Linear Accelerations*

Description

Given a set of locations defining a trajectory, this function computes the linear accelerations between each pair of successive locations along the trajectory.

Usage

```
linear_acc(x, y, t, geo = FALSE)

linAcc(x, y, t, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
t	A vector of timestamps corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the linear accelerations between each pair of successive locations along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

linear_speed, linear_dist

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
t <- as.POSIXct(1:25, origin = Sys.time())
linear_acc(x, y, t)
```

linear_dist *Linear Distances*

Description

Given a set of locations defining a trajectory, this function computes the linear distances between each pair of successive locations along the trajectory.

Usage

```
linear_dist(x, y, geo = FALSE)

linDist(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the linear distances between each pair of successive locations along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

linear_speed, linear_acc, nsd

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
linear_dist(x, y)
```

linear_speed *Linear Speeds*

Description

Given a set of locations defining a trajectory, this function computes the linear speeds between each pair of successive locations along the trajectory.

Usage

```
linear_speed(x, y, t, geo = FALSE)

linSpeed(x, y, t, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
t	A vector of timestamps corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the linear speeds between each pair of successive locations along the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

linear_dist, linear_acc

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
t <- as.POSIXct(1:25, origin = Sys.time())
linear_speed(x, y, t)
```

nn *Nearest Neighbor*

Description

Given the locations of different objects, this function determines the identity of the nearest neighboring object to each object.

Usage

```
nn(x, y, id, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
id	A vector corresponding to the unique identities of each track.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y representing the identity of the nearest neighboring object to each object.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

nnd

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
id <- 1:25
nn(x, y, id)
```

nnd *Nearest Neihgbor Distance*

Description

Given the locations of different objects, this function determines the distance of the nearest neighboring object to each object.

Usage

```
nnd(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y representing the distance to the nearest neighboring object for each object.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

nn

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
id <- 1:25
nnd(x, y)
```

nsd *Net Squared Displacement*

Description

Given a set of locations defining a trajectory, this function computes the net squared displacement of the trajectory, that is the squared distances between each location and the first location of the trajectory

Usage

```
nsd(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates corresponding to a single trajectory.
y	A vector of y (or latitude) coordinates corresponding to a single trajectory.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A vector of the same length as x and y corresponding to the net squared distances between each location and the first location of the trajectory.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

linear_dist

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
nsd(x, y)
```

pdist *Pairwise Distance Matrix*

Description

Given a set of locations, this function computes the distances between each possible pair of locations.

Usage

```
pdist(x, y, geo = FALSE)
```

Arguments

x	A vector of x (or longitude) coordinates.
y	A vector of y (or latitude) coordinates.
geo	A logical value indicating whether the locations are defined by geographic coordinates (pairs of longitude/latitude values). Default: FALSE.

Value

A square matrix representing pairwise distances between each possible pair of locations.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

nn, nnd

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
pdist(x, y)
```

pol_order *Polarization Order Parameter*

Description

Given a set of headings, this function returns the polarization order of the set.

Usage

```
pol_order(h)

polOrder(h)
```

Arguments

h	A vector of headings (in radians).

Value

A single value between 0 and 1 corresponding to the polarization order parameter of the group.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

rot_order

Examples

```
h <- runif(25, 0, 2 * pi)
pol_order(h)
```

sphericity *Sphericity*

Description

Given a set of locations, this function approximates the sphericity of the set by calculating the bivariate 95 the set.

Usage

```
sphericity(x, y)
```

Arguments

x	A vector of x coordinates.
y	A vector of y coordinates.

Value

A single numeric value corresponding to the ratio between the minor and major axis of the bivariate 95 close to 1 indicates that the set is approximately circular; a value close to 0 indicates that the set is strongly elongated.

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

stretch

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
sphericity(x, y)
```

---

stretch *Stretching Direction*

Description

Given a set of locations, this function approximates the stretching direction of the set by calculating the angle of the main axis of the bivariate 95

Usage

```
stretch(x, y)
```

Arguments

x	A vector of x coordinates.
y	A vector of y coordinates.

Value

A single numeric value corresponding to the angle (in radians) of the main axis of the bivariate 95

Author(s)

Simon Garnier, <garnier@njit.edu>

See Also

sphericity

Examples

```
x <- rnorm(25)
y <- rnorm(25, sd = 3)
stretch(x, y)
```

Index

.cartesianPerimeter, 3
.cartesian_perimeter, 4
.ellipse, 4

ang_acc, 5, *6*, *11*
ang_speed, *5*, 6, *11*
angAcc *(*ang_acc*)*, 5
angSpeed *(*ang_speed*)*, 6

centroid, 7, *10*
chull_area, 8, *9*, *12*
chull_perimeter, *4*, *8*, 9, *12*
chullArea *(*chull_area*)*, 8
chullPerimeter *(*chull_perimeter*)*, 9

dist2centroid, *7*, 10

heading, *5, 6*, 11

is_chull, *8, 9*, 12
isChull *(*is_chull*)*, 12

linAcc *(*linear_acc*)*, 13
linDist *(*linear_dist*)*, 14
linear_acc, 13, *14, 15*
linear_dist, *13*, 14, *15*, *18*
linear_speed, *13, 14*, 15
linSpeed *(*linear_speed*)*, 15

nn, 16, *17*, *19*
nnd, *16*, 17, *19*
nsd, *14*, 18

pdist, 19
pol_order, *3*, 20
polOrder *(*pol_order*)*, 20

rot_order, *20*
rot_order *(*.cartesianPerimeter*)*, 3
rotOrder *(*.cartesianPerimeter*)*, 3

sphericity, *5*, 20, *22*
stretch, *5*, *21*, 21
swaRm *(*swaRm-package*)*, 2
swaRm-package, 2