Package ‘swaRmverse’

March 26, 2024

Type Package
Title Swarm Space Creation
Version 0.1.0
Date 2024-02-29
Maintainer Marina Papadopoulou <m.papadopoulou.rug@gmail.com>
Description Provides a pipeline for the comparative analysis of collective movement data (e.g. fish schools, bird flocks, baboon troops) by processing 2-dimensional positional data (x,y,t) from GPS trackers or computer vision tracking systems, discretizing events of collective motion, calculating a set of established metrics that characterize each event, and placing the events in a multi-dimensional swarm space constructed from these metrics. The swarm space concept, the metrics and data sets included are described in: Papadopoulou Marina, Furtbauer Ines, O’Bryan Lisa R., Garnier Simon, Georgopoulou Dimitra G., Bracken Anna M., Christensen Charlotte and King Andrew J. (2023) <doi:10.1098/rstb.2022.0068>.
License GPL-3
Encoding UTF-8
LazyData true
BugReports https://github.com/marinapapa/swaRmverse/issues
RoxygenNote 7.3.1
Depends R (>= 3.5)
Imports parallel, pbapply, Rtsne, trackdf, swaRm, geosphere
Suggests knitr, rmarkdown, ggplot2, testthat (>= 3.0.0), covr
VignetteBuilder knitr
Config/testthat/edition 3
NeedsCompilation no
Author  Marina Papadopoulou [aut, cre]  
(https://orcid.org/0000-0002-6478-8365),  
Simon Garnier [ctb, cph] (https://orcid.org/0000-0002-3886-3974)  

Repository  CRAN  

Date/Publication  2024-03-26 17:00:05 UTC  

R topics documented:  

add_rel_pos_coords .................................................. 3  
add_set_vels .......................................................... 4  
add_velocities .......................................................... 5  
calc_dur_per_event ..................................................... 6  
col_motion_metrics ...................................................... 7  
col_motion_metrics_from_raw ....................................... 9  
define_events .......................................................... 10  
events_dur ............................................................... 11  
events_n ................................................................. 12  
events_summary .......................................................... 13  
event_metrics ............................................................ 14  
expand_pca_swarm_space ............................................. 15  
frontness ............................................................... 16  
get_event_ids ........................................................... 16  
group_metrics .......................................................... 17  
group_metrics_per_set ............................................... 18  
group_shape ............................................................. 20  
moving_average .......................................................... 21  
multi_species_metrics ............................................... 21  
multi_species_pca ...................................................... 22  
multi_species_pca_data ............................................... 23  
new_species_metrics .................................................. 24  
nba ......................................................................... 24  
nn_metrics ............................................................... 25  
normalize_data .......................................................... 26  
pairwise_metrics ......................................................... 27  
pick_threshold ........................................................... 28  
set_data_format .......................................................... 29  
swaRmverse ............................................................... 30  
swarm_space ............................................................. 31  

Index 33
add_rel_pos_coords  

**Relative Position Coordinates**

Description

This function calculates the x and y coordinates of a neighbor in the reference frame of the focal individual.

Usage

```
add_rel_pos_coords(data, focal_heading = c(0, 1))
```

Arguments

- **data**  
  Dataframe with the bearing angle and distance of each individual to specific neighbors. Column names must include: bangl, nnd.

- **focal_heading**  
  The heading of the focal individual, default = c(0,1) for plotting neighbor heading north.

Value

The input dataframe with additional nnx (nearest neighbor x coordinate) and nny (nearest neighbor y coordinate) columns.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

Examples

```
data <- data.frame(  
  bang1 = runif(25, 0, pi),  
  nnd = runif(25)  
)
data <- add_rel_pos_coords(data)
```
Description

This function calculates the headings and speeds of individuals based on two location points and the time taken to travel between those points.

Usage

```r
add_set_vels(data, geo = FALSE, verbose = FALSE, parallelize = FALSE, independent_call = TRUE)
```

Arguments

- `data`: A dataframe with the time series of individuals’ positions. Columns must include: `t`, `id`, `x`, `y`.
- `geo`: Logical, whether positions are geographic coordinates, default = TRUE.
- `verbose`: Logical, whether to post updates on progress, default = FALSE.
- `parallelize`: Logical, whether to run the function in parallel, default = FALSE.
- `independent_call`: Logical, whether the function is called by itself or as part of the package pipeline (through `add_velocities`). The default is set to TRUE, reflecting the direct call of the function by the user.

Value

The input dataframe with a new speed and heading (rotational, in rads) columns.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

- `add_velocities`
**add_velocities**

**Examples**

```r
data <- data.frame(
  x = rnorm(25, sd = 3),
  y = rnorm(25, sd = 3),
  t = as.POSIXct(1:25, origin = Sys.time()),
  id = rep(1, 25)
)

data <- add_set_vels(data, geo = FALSE)
```

**Description**

This function calculates and adds the speed and heading of each individual over time in the dataset, and splits it in a list of dataframes based on the defined sets.

**Usage**

```r
add_velocities(data, geo = FALSE, verbose = FALSE, parallelize = FALSE)
```

**Arguments**

- **data**
  A data frame with time series of individual’s positional data, as exported by the `set_data_format` function. Columns needed: set, t, id, x, y.

- **geo**
  Logical, whether positions are geographic coordinates, default = FALSE.

- **verbose**
  Logical, whether to post updates on progress, default = FALSE.

- **parallelize**
  Logical, whether to run the function in parallel over individuals, default = FALSE.

**Value**

A list of dataframes, an element per set from the input dataframe with new columns: head and speed.

**Author(s)**

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

**See Also**

`add_set_vels`, `set_data_format`
Examples

```r
data <- data.frame(
  set = rep(1, 25),
  x = rnorm(25, sd = 3),
  y = rnorm(25, sd = 3),
  t = as.POSIXct(1:25, origin = Sys.time()),
  id = rep(1, 25)
)

data_list <- add_velocities(data, geo = FALSE)
```

---

calc_dur_per_event  

**Duration of Each Event**

**Description**

This function calculates the duration of each event of collective motion in a dataset.

**Usage**

`calc_dur_per_event(data, step2time)`

**Arguments**

- `data`: A dataframe with an event column (added by `get_event_ids`), indicating the event ID that each timestep belongs to. Timesteps that are not part of an event should not be included in the data.
- `step2time`: The sampling frequency of the dataframe (how many seconds are between each row of the data).

**Value**

A dataframe with two columns, event ID and duration in seconds.

**Author(s)**

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

**See Also**

- `events_dur`
- `get_event_ids`
Examples

data <- data.frame(
    set = c(rep('1', 50), rep('2', 50)),
    event = c(rep(NA, 10), rep(1, 40), rep(2, 30), rep(NA, 20))
)

time_per_row <- 1 # seconds

calc_dur_per_event(data, time_per_row)

Collective Motion Metrics

Description
This function calculates metrics of collective motion across sets and events.

Usage

col_motion_metrics(
    timeseries_data,
    global_metrics,
    step2time = 1,
    verbose = TRUE,
    speed_lim = NA,
    pol_lim = NA,
    noise_thresh = 0
)

Arguments

timeseries_data  A data frame with time series of individual’s positional data through time with nearest neighbor analysis conducted

global_metrics  A data frame with the global metrics timeseries.

step2time  Numeric, the sampling frequency of the dataset (the relation between a time step and real time in seconds).

verbose  Logical, whether to post updates on progress.

speed_lim  Numeric, the threshold of speed for the definition of an event. For more info see: pick_threshold.

pol_lim  Numeric, the threshold of polarization for the definition of an event. For more info see: pick_threshold.

noise_thresh  Numeric, the limit of time difference between consecutive events to be considered the same event. Default value is 0 (no event merging).
Value

A dataframe with metrics of collective motion per event.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

define_events, group_metrics, pairwise_metrics

Examples

```r
## A dataframe with group timeseries
g_df <- data.frame(
  t = as.POSIXct(1:25, origin = "2024-03-18 14:56:05"),
  set = rep(1, 25),
  pol = c(rnorm(25)),
  pol_av = c(rnorm(25)),
  speed = c(rnorm(25)),
  speed_av = c(rnorm(25)),
  shape = c(rnorm(25)),
  event = rep(1, 25),
  N = rep(3, 25)
)

## A dataframe with individual timeseries
p_df <- data.frame(
  t = as.POSIXct(rep(1:25, 3), origin = "2024-03-18 14:56:05"),
  set = rep(1, 75),
  nnd = c(rnorm(75)),
  bangl = runif(75, 0, pi),
  id = c(rep(1, 25), rep(2, 25), rep(3, 25)),
  nn_id = c(
    sample(c(2,3), 25, replace = TRUE),
    sample(c(1,3), 25, replace = TRUE),
    sample(c(2,1), 25, replace = TRUE)),
  event = rep(1, 75)
)
p_df$only_time <- format(p_df$t, "%H:%M:%OS2")

metrics <- col_motion_metrics(
  timeseries_data = p_df,
  global_metrics = g_df,
  step2time = 1,
  speed_lim = 0,
  pol_lim = 0,
  noise_thresh = 1
)
```
Collective Motion Metrics from Raw Data

Description

This function calculates metrics of collective motion across sets and events.

Usage

```r
col_motion_metrics_from_raw(
  data,
  mov_av_time_window,
  step2time = 1,
  geo = FALSE,
  verbose = FALSE,
  speed_lim = NA,
  pol_lim = NA,
  parallelize_all = FALSE,
  noise_thresh = 0
)
```

Arguments

- **data**: A data frame with time series of individual’s positional data through time. Columns must include: `id`, `set`, `t`, `x`, `y`.
- **mov_av_time_window**: Numeric, a time window to average over for speed and polarization timeseries (in timesteps).
- **step2time**: Numeric, the sampling frequency of the dataset (the relation between a time step and real time in seconds).
- **geo**: Logical, whether positions are geographic coordinates, default = FALSE.
- **verbose**: Logical, whether to post updates on progress, default = FALSE.
- **speed_lim**: Numeric, the threshold of speed for the definition of an event. For more info see: `pick_threshold`.
- **pol_lim**: Numeric, the threshold of polarization for the definition of an event. For more info see: `pick_threshold`.
- **parallelize_all**: Logical, whether or not to parallelize over timesteps.
- **noise_thresh**: Numeric, the limit of time difference between consecutive events to be considered the same event. Default value is 0 (no event merging).

Value

A dataframe with metrics of collective motion per event.
define_events

Define Events of Collective Motion

Description

This function adds a keep TRUE/FALSE column in the input dataframe based on whether the average speed and polarization of the group is larger than the input thresholds, reflecting whether a timestep is considered part of a collective event or not.

Usage

define_events(df, sp_lim, pol_lim, step2time, noise_thresh = 0)

Arguments

df
A dataframe with a pol_av and speed_av columns for polarization and speed, respectively (as calculated by the group_metrics_per_set function).

sp_lim
The (lower) threshold of speed to use for defining which timesteps are part of an events of collective motion. In other words, during an event the group should have an average speed of at least sp_lim.
The (lower) threshold of polarization to use for defining which timesteps are part of an events of collective motion. In other words, during an event the group’s polarization should be at least \texttt{pol_lim}.

\texttt{step2time}  
Sampling frequency, i.e. the relation between time steps (rows) in the input dataframe and real time (in seconds).

\texttt{noise_thresh}  
The limit of time difference between consecutive events to be considered the same event. The default value is 0 (no event merging).

Value
the dataframe that was given as input with an extra \texttt{keep} column. The function also prints the number and duration of the defined events.

\textbf{Author(s)}
Marina Papadopoulou \texttt{<m.papadopoulou.rug@gmail.com>}

\textbf{See Also}
\texttt{pick_threshold, group_metrics_per_set}

\textbf{Examples}
```
data <- data.frame(
  set = rep(’1’, 50),
  pol_av = rnorm(50, mean = 0.5, sd = 0.2),
  speed_av = rnorm(50, mean = 5)
)
data <- define_events(data, sp_lim = 5, pol_lim = 0.4, step2time = 1)
```

---

\textbf{events_dur} \hspace{1cm} \textit{Total Duration of All Events}

\textbf{Description}
This function calculates the total duration (in seconds) of events of collective motion in a dataset.

\textbf{Usage}
\texttt{events_dur(data, step2time)}

\textbf{Arguments}
\texttt{data} \hspace{1cm} A dataframe with a \texttt{keep} column, representing which rows are defined as events of collective motion (added by the \texttt{define_events} function).

\texttt{step2time} \hspace{1cm} The sampling frequency of the dataframe (how many seconds are between each row of the data).
Value

A numeric corresponding to the total duration of events in the dataset in seconds.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

`define_events`, `events_n`

---

<table>
<thead>
<tr>
<th>events_n</th>
<th>Number of Events</th>
</tr>
</thead>
</table>

Description

This function calculates the number of events of collective motion in a dataset.

Usage

`events_n(data)`

Arguments

data

A dataframe with a `keep` column (representing which rows are defined as events of collective motion) and a `set` column.

Value

an integer with the number of events of collective motion (sequences of `keep == TRUE`).

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

`define_events`

Examples

```r
data <- data.frame(
  set = c(rep('1', 50), rep('2', 50)),
  keep = c(rep(FALSE, 10), rep(TRUE, 70), rep(FALSE, 20))
)

events_n(data) # 2 events
```
Description

This function summarizes the number of events and their total duration in the dataset.

Usage

events_summary(data, step2time)

Arguments

data A dataframe with a keep column (representing which rows are defined as events of collective motion) and a set column.

step2time The sampling frequency of the dataframe (how many seconds are between each row of the data).

Value

A dataframe with 3 columns: set, ev_count (number of events), and dur (duration of events in seconds).

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

events_dur, events_n

Examples

data <- data.frame(
  set = c(rep('1', 50), rep('2', 50)),
  keep = c(rep(FALSE, 10), rep(TRUE, 70), rep(FALSE, 20))
)

time_per_row <- 1 # seconds

events_summary(data, time_per_row)
event_metrics

Metrics of Collective Motion

Description
This function calculates metrics of collective motion across sets and events.

Usage

```r
event_metrics(global_df, pairwise_df)
```

Arguments

- `global_df`: A data frame with time series of global group measurements. Columns must include: `set`, `t`, `event`, `pol`, `shape`, `speed`.
- `pairwise_df`: A data frame with time series of pairwise measurements. Columns must include: `set`, `t`, `id`, `nnd`, `bangl`.

Value
A dataframe with 10 metrics per event.

Author(s)
Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

- `group_metrics`, `nn_metrics`

Examples

```r
## A dataframe with group timeseries
g_df <- data.frame(
t = 1:25,
set = rep(1, 25),
pol = c(rnorm(25)),
speed = c(rnorm(25)),
shape = c(rnorm(25)),
etvent = rep(1, 25),
N = rep(2, 25)
)

## A dataframe with individual timeseries
p_df <- data.frame(
t = rep(1:25, 2),
set = rep(1, 50),
nnd = c(rnorm(50)),
```
expand_pca_swarm_space

```r
bangl = runif(25, 0, pi),
id = c(rep(1, 25), rep(2, 25)),
event = rep(1, 50)
)

events_dataframe <- event_metrics(g_df, p_df)
```

---

**Expand Existing Swarm Space (PCA)**

**Description**

This function predicts the positions of new event data in an existing PCA space using the `stats::predict` function.

**Usage**

```r
expand_pca_swarm_space(metrics_data, pca_space, event_dur_limit = NA)
```

**Arguments**

- `metrics_data` A dataframe with the new metrics data to add in swarm space.
- `pca_space` The PCA object to predict from, the output of the `stats::prcomp` function or the pca element of the list output of the `swarm_space` function.
- `event_dur_limit` Numeric, capturing an event duration value in seconds. Used to filter out events that are shorter that this value. Default = NA, no filtering is applied.

**Value**

A dataframe with the x and y coordinates in the input swarm space per event of the new species.

**Author(s)**

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

**See Also**

- `swarm_space`

**Examples**

```r
data(multi_species_metrics)
data(multi_species_pca)
ss <- expand_pca_swarm_space(multi_species_metrics, multi_species_pca)
```
**Frontness**

Description
Given the bearing angle of an object to another, this function calculates the frontness, a value that ranges from 0 to 1 and represents how in front the focal object is from its neighbor.

Usage
```
frontness(bs)
```

Arguments
- `bs`: A vector of bearing angles (in rad) between objects.

Value
A vector of the same length as `bs` representing the frontness of a focal object to its neighbor.

Author(s)
Marina Papadopoulou, <m.papadopoulou.rug@gmail.com>

See Also
- `nnba`

Examples
```
bs <- runif(25, max = pi)
frontness(bs)
```

**get_event_ids**

Description
This function returns a vector with the timeseries of event IDs according to the input keep column of the dataframe.

Usage
```
get_event_ids(df)
```
Arguments

df  A dataframe with a set and a keep column to get the timeseries of event IDs. The keep column is added by the `define_events` function and represents whether each timestep is part of an event or not (whether it should be kept in for the rest of the analysis). Each set of the dataframe should be ordered in time.

Value

a vector of the same length as the rows of the input dataframe with the timeseries of event IDs.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

define_events

Examples

data <- data.frame(
  set = c(rep('1', 50), rep('2', 50)),
  keep = c(rep(FALSE, 10), rep(TRUE, 70), rep(FALSE, 20))
)
data$event <- get_event_ids(data)

Description

This function calculates the average speed, polarization and shape of a group through time.

Usage

group_metrics(data, geo, step2time = 1, parallelize = FALSE)

Arguments

data  A dataframe of (ordered) time series of headings, positions, and speeds per individual. The dataframe may contain several individuals. Should include the columns: id, t, speed, x, y, head, set.
geo  Logical, whether positions are geographic coordinates, default = FALSE.
step2time  Numeric, the sampling frequency of the data (the relation between a row in the data and real time in seconds).
parallelize  Logical, whether to parallelize over time. Suggested only for very large datasets.
Value

A dataframe with the group average timeseries, with columns: set, t, pol, speed, shape, N (number of individuals), missing_ind (whether some individuals are missing).

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

group_shape, add_velocities

Examples

data <- data.frame(
  set = rep("1", 50),
  t = as.POSIXct(rep(1:25, 2), origin = Sys.time()),
  id = c(rep(1, 25), rep(2, 25)),
  x = rnorm(50),
  y = rnorm(50),
  head = runif(50, 0, 2 * pi),
  speed = rnorm(50)
)

gm <- group_metrics(data,
  geo = FALSE,
  step2time = 1)

---

**group_metrics_per_set**  Group Metrics of Collective Motion in a Dataset

Description

This function calculates the timeseries of average speed, polarization and shape of all set in a dataset

Usage

group_metrics_per_set(
  data_list,
  mov_av_time_window,
  geo,
  step2time,
  parallelize = FALSE
)
Arguments

data_list A list of dataframes with groups timeseries per set. Columns must include: id, t, set, head, x, y, speed.

mov_av_time_window Integer, timesteps to use as a sliding window for average speed and polarization.

geo Logical, whether positions are geographic coordinates, default = FALSE.

step2time Double, the sampling frequency of the data (the relation between a time step and real time in seconds).

parallelize Logical, whether or not to parallelize over the timesteps of each set.

Value

A dataframe with the group average timeseries for each set, with columns: set, t, pol, speed, shape, N (number of individuals), missing_ind (whether some individuals are missing), pol_av (moving average of polarization based on input time window) and speed_av (moving average of speed based on input time window).

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

group_metrics, moving_average

Examples

data <- data.frame(
  set = rep("1", 50),
  t = as.POSIXct(rep(1:25, 2), origin = Sys.time()),
  id = c(rep(1, 25), rep(2, 25)),
  x = rnorm(50),
  y = rnorm(50),
  head = runif(50, 0, 2 * pi),
  speed = rnorm(50)
)

gm <- group_metrics_per_set(list(data),
  mov_av_time_window = 5,
  geo = FALSE,
  step2time = 1
)
group_shape

**Group Shape Based on a OOB**

**Description**

Calculates how oblong the shape of a group is, relative to its average moving direction, along with the properties of the minimum object oriented bounding box (OBBB) around all objects.

**Usage**

```r
group_shape(x, y, hs, geo = FALSE)
```

**Arguments**

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

**Value**

A list with the estimate of how oblong the group is, and the details of the bounding box, i.e. its coordinates, height, width, and orientation of its longest side in degrees.

**Author(s)**

Marina Papadopoulou, <m.papadopoulou.rug@gmail.com>

**Examples**

```r
x <- rnorm(25)
y <- rnorm(25, sd = 3)
h <- runif(25, 0, 2 * pi)
group_shape(x, y, h, geo = FALSE)
```
moving_average

Moving Average

Description

This function calculates the moving average of a time series.

Usage

moving_average(timeseries, window)

Arguments

timeseries Vector of doubles representing a timeseries.
window Double, the time-window to average over (in timesteps).

Value

A vector of doubles (average over the window).

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

Examples

bs <- rnorm(20, mean = 10, sd = 1)
moving_average(bs, 5)

multi_species_metrics Multi-Species Collective Motion Metrics

Description

A dataset containing the metrics of collective motion for 4 species: stickleback fish, homing pigeons, goats, and chacma baboons. They were used for the construction of the initial swarm space in:


Usage

data('multi_species_metrics')
**multi_species_pca**

**Format**

A dataframe with 118 rows and 12 columns:

- **mean_mean_nnd** Average nearest neighbor distance
- **mean_sd_nnd** Average within-group variation in nearest neighbor distance
- **sd_mean_nnd** Temporal variation in average nearest neighbor distance
- **mean_pol** Average polarization
- **sd_pol** Temporal variation in polarization
- **stdv_speed** Temporal variation in speed
- **mean_sd_front** Average within-group variation in frontness
- **mean_mean_bangl** Average bearing angle
- **mean_shape** Average group shape (rads)
- **sd_shape** Temporal variation in group shape (rads)
- **species** Species id
- **event** Event id

**References**


---

**multi_species_pca**  
**Multi-Species PCA**

**Description**

The swarm space PCA of 4 species: stickleback fish, homing pigeons, goats and chacma baboons. First published as part of:


**Usage**

data('multi_species_pca')

**Format**

A list of 5 elements, exported by the stats::prcomp function.
multi_species_pca_data

References


See Also

multi_species_pca_data

multi_species_pca_data

Multi-Species PCA Data

Description

The positions of events from 4 species: stickleback fish, homing pigeons, goats and chacma baboons, in the PCA swarm space (see multi_species_pca. First published as part of:


Usage

data('multi_species_pca_data')

Format

A dataframe of 3 columns: species, PC1, PC2, PC3.

References


See Also

multi_species_pca
new_species_metrics  The Collective Motion Metrics of a New Species

Description
The output dataset of vignette 2, containing the metrics of collective motion for a new species.

Usage
data('new_species_metrics')

Format
An object of class data.frame with 60 rows and 16 columns.

nnba  Bearing Angle to Nearest Neighbor

Description
Given the locations and headings of different objects, this function determines the angle between the heading of each object and the position to the nearest neighboring object.

Usage
nnba(x, y, hs, geo = FALSE)

Arguments
x  A vector of x (or longitude) coordinates.
y  A vector of y (or latitude) coordinates.
hs  A vector of headings (angle in rads).
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, Marina Papadopoulou, m.papadopoulou.rug@gmail.com
nn_metrics

See Also

pdist

Examples

x <- rnorm(25)
y <- rnorm(25, sd = 3)
hs <- rnorm(25, sd = 1)
nxba(x, y, hs)

nn_metrics

Nearest Neighbour Metrics

Description

This function calculates the bearing angle and distance from all focal individuals in a group to their nearest neighbor over time.

Usage

nn_metrics(
  data,
  add_coords = FALSE,
  geo = FALSE,
  verbose = FALSE,
  parallelize = FALSE
)

Arguments

data
  A dataframe with the group’s positional timeseries for one set. Column names must include: id, t, head, x, y. The calculations are based on the swaRm package.

add_coords
  Logical, whether the data on relative positions of nearest neighbours should be converted into coordinates in the reference frame of the focal individual (nnx, nny). This can be useful for visualization purposes but it is not used in the package pipeling. Default = ‘FALSE’.

geo
  Logical, whether positions are geographic coordinates, default = FALSE.

verbose
  Logical, whether to post updates on progress.

parallelize
  Logical, whether to parallelize the function over time.

Value

The input dataframe with new columns for nearest neighbor id (nn_id), bearing angle (bangl), and distance (nnd). If add_coords is TRUE, the columns nnx and nny are added.
**normalize_data**

**Normalize Data**

This function rescales a vector to values between 0 and 1.

**Usage**

```r
normalize_data(vec)
```

**Arguments**

- `vec` A numerical vector to normalize.

**Value**

A vector of doubles, the normalized values of the input vector.

**Author(s)**

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

**Examples**

```r
d <- rnorm(20, mean = 10, sd = 1)
normalize_data(d)
```
pairwise_metrics  

Pairwise Metrics of Collective Motion in a Dataset

Description

This function calculates the bearing angle and distance from each focal individual of a group to its nearest neighbor over time, across the sets of a dataset.

Usage

```r
pairwise_metrics(
  data_list,
  geo = FALSE,
  verbose = FALSE,
  parallelize = FALSE,
  add_coords = FALSE
)
```

Arguments

- **data_list**: A list of dataframes with groups timeseries per set. Columns must include: id, t, set, head, x, y.
- **geo**: Logical, whether positions are geographic coordinates, default = FALSE.
- **verbose**: Logical, whether to post updates on progress, default = FALSE.
- **parallelize**: Logical, whether to run the function in parallel over timesteps, default = FALSE.
- **add_coords**: Logical, whether data on relative positions are converted into geographic coordinates, default = 'FALSE'.

Value

A dataframe format of the input list, with new columns for nearest neighbor id (nn_id), bearing angles (bangl), and distances (nnd). If add_coords is TRUE, the columns nnx and nny are also added.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

- `nn_metrics`, `group_metrics_per_set`
Examples

data <- data.frame(
  set = rep("1", 50),
  t = as.POSIXct(rep(1:25, 2), origin = Sys.time()),
  id = c(rep(1, 25), rep(2, 25)),
  x = rnorm(50),
  y = rnorm(50),
  head = runif(50, 0, 2 * pi)
)

pm <- pairwise_metrics(list(data), geo = FALSE)

pick_threshold

Description

An interactive function that calculates and prints the quantiles of the input distribution and asks the user to input the threshold value they want to keep. If a threshold is given as input, then the function checks that the threshold type is correct and returns it. In the swaRmverse framework, the timesteps with lower values than the threshold will be labelled as not part of an event.

Usage

pick_threshold(data_distr, var, threshold = NA)

Arguments

data_distr  A numeric vector to pick a threshold for. In the package's pipeline it is the timeseries of polarization and average speed of a group.

var  A string, the of the distribution to use at the interactive step to ask the user for input.

threshold  If NA (the default), the function runs in interactive mode and the user inputs a given value to return. If numeric, the function just returns this input (interactive case is off).

Value

the selected or input value of the user for the lower threshold, of the variable to be used for the definition of an event.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>
See Also

`define_events`

Examples

```r
d <- rnorm(25, sd = 1)
d_variable_name <- "a variable"
the_threshold <- 0
pick_threshold(d, d_variable_name, threshold = the_threshold)

## If the threshold is not known, run the interactive version
## without giving a threshold as input.
```

---

**Description**

This function is a wrapper for the `track` function of the `trackdf` package.

**Usage**

```r
set_data_format(
  raw_x,
  raw_y,
  raw_t,
  raw_id,
  origin,
  period,
  tz,
  proj,
  format,
  ...
)
```

**Arguments**

- `raw_x` A numeric vector representing the x coordinates of individual(s).
- `raw_y` A numeric vector representing the y coordinates of individual(s).
- `raw_t` A numeric vector that can be coerced to date-time objects by `as_datetime` representing the times (or frames) at which each location was recorded.
- `raw_id` A vector representing the identity of each coordinate recording.
- `origin` Something that can be coerced to a date-time object by `as_datetime` representing the start date and time of the observations when `t` is a numeric vector.
- `period` A character vector in a shorthand format (e.g. "1 second") or ISO 8601 specification. This is used when `t` is a numeric vector to represent time unit of the observations.
tz A time zone name. See OlsonNames.

proj A character string or a sp::CRS object representing the projection of the coordinates. Leave empty if the coordinates are not projected (e.g., output of video tracking). "+proj=longlat" is suitable for the output of most GPS trackers.

format A character string indicating the formatting of ‘t’. See strptime for how to specify this parameter.

... Additional vectors representing categories that the data should be split by. If none, only the date will be used as a unit of data separation.

Value
A track dataframe table, which is a colloquial term for an object of class track.

Author(s)
Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

Examples

```r
raw_data <- data.frame(
  frame = rep(1:25, 3),
  x = rnorm(75),
  y = rnorm(75),
  id = c(rep(1, 25), rep(2, 25), rep(3, 25))
)
data <- set_data_format(
  raw_x = raw_data$x,
  raw_y = raw_data$y,
  raw_t = raw_data$frame,
  raw_id = raw_data$id,
  period = 1,
  origin = Sys.time(),
  tz = "Africa/Windhoek"
)
```

Description
The swaRmverse package provides a pipeline for the comparative analysis of collective movement data (e.g. fish schools, bird flocks, baboon troops) by processing 2-dimensional positional data (x,y,t) from GPS trackers or computer vision tracking systems, discretizing events of collective motion, calculating a set of established metrics that characterize each event, and placing the events in a multi-dimensional ‘swarm space’ constructed from these metrics.
## Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>swaRmverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>0.1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2024-02-29</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-3</td>
</tr>
</tbody>
</table>

## Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>
Simon Garnier <garnier@njit.edu>
Maintainer: Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

## References


## See Also

Useful links:

- [https://marinapapa.github.io/swaRmverse/](https://marinapapa.github.io/swaRmverse/)
- [https://github.com/marinapapa/swaRmverse](https://github.com/marinapapa/swaRmverse)
- Report bugs at [https://github.com/marinapapa/swaRmverse/issues](https://github.com/marinapapa/swaRmverse/issues)

---

### swarm_space

Create a Swarm Space

## Description

This function runs a PCA (Principal component analysis) or a t-SNE (t-distributed Stochastic Neighbor Embedding) over the global and pairwise metrics of collective motion per each event to produce a swarm space. The PCA is computed with the `stats::prcomp` function and the t-SNE with the `Rtsne::Rtsne` function.
Usage

```r
swarm_space(
  metrics_data,
  space_type = "pca",
  event_dur_limit = NA,
  tsne_rand_seed = NA,
  tsne_perplexity = 25
)
```

Arguments

- **metrics_data**: A dataframe with metrics of collective motion per event.
- **space_type**: A string, stating the choice between PCA ("pca") and t-SNE ("tsne"), default = "pca".
- **event_dur_limit**: Numeric, capturing an event duration value in seconds. Used to filter out events that are shorter that this value. Default = NA, no filtering is applied.
- **tsne_rand_seed**: Numeric, the random seed for the t-SNE analysis, to ensure reproducibility. Default = NA, but a value should be given if the t-SNE analysis is selected.
- **tsne_perplexity**: Numeric, the perplexity parameter for the t-SNE analysis. Usually between 10-50, default = 25.

Value

A list with 3 elements: a dataframe representing the swarm space (x and y coordinates per event of each species), a reference dataframe (ref) including all the additional event information from the input metric data dataframe, a dataframe for the t-SNE analysis (tsne_setup) that includes the input parameters used, and a list for the PCA analysis (pca) with the output of the `stats::prcomp` command.

Author(s)

Marina Papadopoulou <m.papadopoulou.rug@gmail.com>

See Also

`group_metrics, pairwise_metrics, nn_metrics, col_motion_metrics`

Examples

```r
data(multi_species_metrics)
ss <- swarm_space(multi_species_metrics)
```
Index

* datasets
  - multi_species_metrics, 21
  - multi_species_pca, 22
  - multi_species_pca_data, 23
  - new_species_metrics, 24

add_rel_pos_coords, 3, 26
add_set_vels, 4, 5
add_velocities, 4, 5, 10, 18

calc_dur_per_event, 6
col_motion_metrics, 7, 32
col_motion_metrics_from_raw, 9

define_events, 8, 10, 11, 12, 17, 29

event_metrics, 14
events_dur, 6, 11, 13
events_n, 12, 12, 13
events_summary, 13
expand_pca_swarm_space, 15

frontness, 16

generate_event_ids, 6, 16
group_metrics, 8, 10, 14, 17, 19, 26, 32
group_metrics_per_set, 10, 11, 18, 27
group_shape, 18, 20

moving_average, 10, 19, 21
multi_species_metrics, 21
multi_species_pca, 22, 23
multi_species_pca_data, 23, 23

new_species_metrics, 24
nn_metrics, 14, 25, 27, 32
nnba, 16, 24
normalize_data, 26

pairwise_metrics, 8, 10, 27, 32
pdist, 25

pick_threshold, 7, 9, 11, 28

set_data_format, 5, 29
swarm_space, 15, 31
swarmverse, 30
swarmverse-package (swarmverse), 30