Package: wildlifeDI (via r-universe)

August 28, 2024

Type Package

Title Calculate Indices of Dynamic Interaction for Wildlife Tracking Data

Version 1.0.1

Description Dynamic interaction refers to spatial-temporal associations in the movements of two (or more) animals. This package provides tools for calculating a suite of indices used for quantifying dynamic interaction with wildlife telemetry data. For more information on each of the methods employed see the references within. The package (as of version >= 0.3) also has new tools for automating contact analysis in large tracking datasets. The package (as of version 1.0) uses the 'move2' class of objects for working with tracking dataset.

URL https://github.com/jedalong/wildlifeDI

Depends R (>= 3.5.0)

Suggests ggplot2, knitr, rmarkdown, igraph

Imports move2, adehabitatLT, sp, sf, stats, graphics, dplyr, units, lwgeom

License GPL-3

VignetteBuilder knitr

RoxygenNote 7.3.1

Encoding UTF-8

Repository https://jedalong.r-universe.dev

RemoteUrl https://github.com/jedalong/wildlifedi

RemoteRef HEAD

RemoteSha 91b38fec0469d6de589b3ec683288235ba2f840c

Contents

Ca 2

checkTO 3
conDisplacement 4
conPhase 6
conProcess 7
conTimelag 8
Cr 9
Cs 10
dcPlot 12
deer 13
DI 14
does 15
Don 16
GetSimultaneous 17
GetTO 18
HAI 19
IAB 20
Lixn 22
ltraj_move2 24
move2_ltraj 25
Prox 26

Index **28**

`Ca` *Coefficient of Association*

Description

This function measures the dynamic interaction between two moving objects following the methods first described by Cole (1949), and more recently employed by Bauman (1998).

Usage

```
Ca(traj, traj2, tc = 0, dc = 0)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	temporal tolerance limit (in seconds) for defining when two fixes are simultaneous or together. Parameter passed to function `GetSimultaneous`.
`dc`	distance tolerance limit (in appropriate units) for defining when two fixes are spatially together.

Details

This function can be used to calculate the Cole (1949) measure of dynamic interaction between two animals. Termed a coefficient of association, the Ca statistic tests the number of fixes the animals are observed together against the total number of fixes following:

$$Ca = \frac{2AB}{A + B}$$

where A (respectively B) is the number of times animal 1 (resp. 2) are observed, and AB is the number of times the two animals are observed together. Several works, including Bauman (1998) have suggested that Ca > 0.5 indicates affiliation or fidelity, while Ca < 0.5 indicates no association between the two animals. Note that this function calls `GetSimultaneous` to identify the temporal component of identifying when fixes together.

Value

This function returns a numeric result of the Ca statistic for each pair in the dataset.

References

Bauman, P.J. (1998) The Wind Cave National Park elk herd: home ranges, seasonal movements, and alternative control methods. M.S. Thesis. South Dakota State University, Brookings, South Dakota, USA.

Cole, L.C. (1949) The measurement of interspecific association. *Ecology*. **30**, 411–424.

See Also

GetSimultaneous, Prox, HAI

Examples

```
data(deer)
#tc = 7.5 minutes, dc = 50 meters
Ca(deer, tc = 7.5*60, dc = 50)
```

`checkTO` *Check for temporal overlap*

Description

The function `checkTO` is a simple function for identifying if, and for how long, two telemetry datasets overlap (temporally) with each other. The function returns a dataframe with5 columns of information: the ids of the first an second individuals in a dyad, a logical variable indicating if the two trajectories overlap temporally, and timings of the beginning and end of the overlap period. If only a single move2 object is provided it considers all pairwise dyads within that move2 object. If two move2 objects are passed in it considers only the dyad pairs from traj against traj2. This can be used to test only the pairwise dyads between two groups (e.g., inter-species).

Usage

```
checkTO(traj, traj2)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) an object of the class `move2` which contains the time-stamped movement fixes. For more information on objects of this type see `help(mt_as_move2)`.

Details

The function `checkTO` can be used to identify if, when, and for how long the tracking data of two individuals overlap temporally.

Value

A `data.frame` of with five columns, ID1, ID2, TO (logical indicating if the two tracking dataset overlap temporally), the beginning (`tmin`), and end (`tmax`) of the overlap period, stored as `POSIX` objects.

See Also

GetSimultaneous, GetTO

Examples

```
data(does)
dyads <- checkTO(does)
```

`conDisplacement` *Calculate net displacement from contacts*

Description

Calculate the net-displacement (distance) of fixes before and after a contact phase from the nearest contact pahse in time.

Usage

```
conDisplacement(traj, def = "all")
```

Arguments

`traj`	an object of the class `move2` which should be output from the function `conPhase`.
`def`	how to define the point-of-contact. The default is to define it as all fixes in a phase `type = 'all'`, alternatively contacts can be defined as a single point along the phase defined as one of: `'first','last','minDist','minTime'`, which corresponds to the first fix int he contact phase, the last fix in the contact phase, the fix with the minimum time difference and the fix with the closest contact distance.

Details

This function is used to compute the net displacement away from contacts by an animal before and after a contact phase. Net displacement represents an important variable related to the movement of the individual.

Value

An move2 object with a new 'contact_displacement' column indicating the straight-line distance to the nearest (in time) contact phase (defined using parameter def). If there are no contacts associated with an individual the contact displacement is NA.

References

Long, JA, Webb, SL, Harju, SM, Gee, KL (2022) Analyzing Contacts and Behavior from High Frequency Tracking Data Using the wildlifeDI R Package. *Geographical Analysis*. **54**, 648–663.

See Also

conProcess, conPhase, conTimelag

Examples

```
## Not run:
data(does)
doecons <- conProcess(does,tc=15*60,dc=50)
doephas <- conPhase(doecons,pc=60*60)
disp_f <- conDisplacement(doephas,def='first')
disp_l <- conDisplacement(doephas,def='last')

## End(Not run)
```

`conPhase` *Process contact phases*

Description

Computes phases where contacts occur based on a temporal tolerance.

Usage

```
conPhase(traj, pc = 0)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`. This will be output from the function `conProcess`.
`pc`	time (in seconds) to allow for which to combine contact events (see details).

Details

This function is used following the `conProcess` function to arrange contacts into phases where continuous contact occurs (based on the user-defined time threshold `pc`. The idea is that we can consider a phase to be a continuous contact event (based on `dc` see `conProcess`) as long as the contact is only interrupted for no more than `pc` time units.

Value

An `move2` object with new column `contact_pha`.

References

Long, JA, Webb, SL, Harju, SM, Gee, KL (2022) Analyzing Contacts and Behavior from High Frequency Tracking Data Using the wildlifeDI R Package. *Geographical Analysis*. **54**, 648–663.

See Also

conProcess, conSpatial, conTemporal, conSummary

Examples

```
## Not run:
data(does)
doecons <- conProcess(does,tc=15*60,dc=50)
doephas <- conPhase(doecons,pc=60*60)

## End(Not run)
```

`conProcess` *Process contacts*

Description

This function performs basic contact analysis between individuals in a group of tracked animals, or between two different groups of tracked animals.

Usage

```
conProcess(traj, traj2, dc = 0, tc = 0, GetSim = TRUE, fixid, return = "move2")
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`. If traj2 is specified traj may have only one individual.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`dc`	distance tolerance limit (in appropriate units) for defining when two fixes are spatially together.
`tc`	time threshold for determining simultaneous fixes – see function: GetSimultaneous.
`GetSim`	(logical) whether or not to use GetSimultaneous to time match fixes between pairs of individuals. Default = TRUE.
`fixid`	(optional) a column providing unique fix ID's or if not specified one is created by combining the track ID with the fix number for that individual (e.g., "Leroy_107").
`return`	What to return (one of 'move2' (default) or 'contacts'). See Return below.

Details

This function can be used to identify all fixes defined as contacts in space and time between individuals in one or two groups.

Value

If return = 'move2' (the default) this function returns the input traj move2 object with additional columns: contact - (binary) whether or not a fix is a contact, contact_id - the id of the individual with which a contact occurs, contact_d - the proximity distance of the contact, contact_dt - the difference in time between the two fixes in the contact, contact_n - the number of contacts at that time. In the event that there is more than one contact for a given fix, the contact_id, contact_d, and contact_dt values are all associated with the most proximal (in geographical space) contact. If return = 'contacts' this function returns a data.frame with the columns: (id1,id2) the id's of the individuals involved in a contact, the unique fix id's from the original data associated with each of the fixes involved in a contact (see parameter fixid), the times of the contact fixes, (dist) the distance between the two fixes associated with the contact, and (difftime) the difference in time between the two fixes involved in the contact.

References

Long, JA, Webb, SL, Harju, SM, Gee, KL (2022) Analyzing Contacts and Behavior from High Frequency Tracking Data Using the wildlifeDI R Package. *Geographical Analysis*. **54**, 648–663.

See Also

GetSimultaneous, dcPlot, conPhase

Examples

```
## Not run:
data(does)
doecons <- conProcess(does,tc=15*60,dc=50)

## End(Not run)
```

`conTimelag` *Compute time-lags from contact phases*

Description

Computes the time-lag from the nearest contact phase.

Usage

```
conTimelag(traj, def = "all")
```

Arguments

`traj`	an object of the class `move2` which should be output from the function `conPhase`.
`def`	how to define the point-of-contact. The default is to define it as all fixes in a phase `def = 'all'`, alternatively contacts can be defined as a single point along the phase defined as one of: `'first'`,`'last'`,`'minDist'`,`'minTime'`, which corresponds to the first fix int he contact phase, the last fix in the contact phase, the fix with the minimum time difference and the fix with the closest contact distance.

Details

This function is used following the `conphase` function. One should choose how to define the contact point (i.e., the parameter `contact`) depending on the research question.

Value

A move2 object with an additional column contact_timelag with the time to the nearest (in time) contact phase. Negative values indicate times prior to the nearest contact phase and postive values indicate times after the nearest contact phase. If an individual has no contacts, the contact time-lag is NA.

References

Long, JA, Webb, SL, Harju, SM, Gee, KL (2022) Analyzing Contacts and Behavior from High Frequency Tracking Data Using the wildlifeDI R Package. *Geographical Analysis*. **54**, 648–663.

See Also

conPhase, conProcess, conDisplacement

Examples

```
## Not run:
data(does)
doecons <- conProcess(does,tc=15*60,dc=50)
doephas <- conPhase(doecons,pc=60*60)
conTL_first <- conTimelag(doephas,def='first')
conTL_all <- conTimelag(doephas,def='all')

## End(Not run)
```

`Cr` *Movement Correlation Coefficient*

Description

The function `Cr` computes the correlation statistic for movement data as presented in the paper by Shirabe (2006). The statistic is essentially a Pearson product-moment correlation statistic formulated for use with movement data.

Usage

```
Cr(traj, traj2, tc = 0)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.

Details

The function `Cr` can be used to measure the level of dynamic interaction (termed correlation) between a pair of simultaneously moving objects. The statistic is sensitive to interaction in both movement direction (azimuth) and displacement, but is unable to disentangle the effects of these components. NOTE: This function is only appropriate with projected coordinates.

Value

This function returns the Shirabe (2006) correlation statistic for two moving objects.

References

Shirabe, T. 2006. Correlation analysis of discrete motions. In: Raubal, M., Miller, HJ, Frank, AU, and Goodchild, M. eds. GIScience 2006, LNCS 4197. Berlin: Springer-Verlag; 370-382.

See Also

GetSimultaneous, DI

Examples

```
data(deer)
#tc = 7.5 minutes
Cr(deer, tc = 7.5*60)
```

`Cs` *Coefficient of Sociality*

Description

The function `Cs` computes the coefficient of sociality between two moving objects following the methods outlined by Kenward et al. (1993). It also uses a signed Wilcoxon-rank test to test for significance.

Usage

```
Cs(traj, traj2, tc = 0)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.

Details

This function can be used to calculate the Kenward et al. (1993) coefficient of sociality (Cs) between two animals. The Cs statistic tests the observed mean distance between simultaneous fixes against that expected by the overall distribution of distances between all fixes.

$$Cs = \frac{D_E - D_O}{D_O + D_E}$$

Where D_O is the mean observed distance between simultaneous fixes, and D_E is the mean expected distance between all fixes. Kenward et al. (1993) propose Cs as a useful metric for exploring attraction or avoidance behaviour. Values for Cs closer to 1 indicate attraction, while values for Cs closer to -1 indicate avoidance. Values of Cs near 0 indicate that the two animals' movements have no influence on one another.

Further, the difference between the observed and expected distances are compared using a paired signed-rank test (both one-sided tests, indicative of attraction or avoidance). See the function `GetSimultaneous` for details on how simultaneous fixes are determined from two trajectories.

Value

This function returns a list of objects representing the calculated values from the Cs statistic and associated *p*-values from the signed rank test.

- Do – The mean distance of simultaneous fixes.
- De – The mean expected distance, from all fixes.
- Cs – The coefficient of sociality, see **Details**.
- p.Attract – One sided *p*-value from signed rank test, testing for attraction.
- p.Avoid – One sided *p*-value from signed rank test, testing for avoidance.

References

Kenward, R.E., Marcstrom, V. and Karlbom, M. (1993) Post-nestling behaviour in goshawks, *Accipiter gentilis: II.* Sex differences in sociality and nest-switching. *Animal Behaviour.* **46**, 371–378.

See Also

GetSimultaneous

Examples

```
data(deer)
#tc = 7.5 minutes
Cs(deer, tc = 7.5*60)
```

`dcPlot` *Contact distance plot*

Description

This function is an exploratory tool to examine the pairwise distances between individuals within a large telemetry dataset.

Usage

```
dcPlot(traj, traj2, tc = 0, histplot = TRUE, dmax)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`.
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`histplot`	(logical) whether to output a histogram, along with a list of the natural breaks in the histogram (`histplot = TRUE`) or the dataframe of all paired distances used to construct the histogram (`histplot=FALSE`) to be used for further analysis.
`dmax`	(optional) distance value to 'cut-off' the distance histogram.

Details

The dcPlot function can be used to study the frequency distribution of pairwise distances between individual in a large telemetry dataset. It can be applied to a single group (if `mtraj2` is ignored) or two-groups of individuals. The code attempts to find natural breaks (local minima) in the frequency histogram using an approach based on the `peaks` function attributed to B. Ripley (see `https://stackoverflow.com/questions/6324354/add-a-curve-that-fits-the-peaks-from-a-plot-in-r`). This tool is meant to be used for exploratory data analysis.

Value

If `histplot = TRUE` a list of the natural breaks (local minima) identified from the frequency histogram and a plot of the frequency histogram. If `histplot = FALSE` a dataframe containing all the pairwise and simultaneous distances between all individuals in the trajectory dataset.

See Also

GetSimultaneous, conProcess, Prox, Don, IAB

Examples

```
## Not run:
data(does)
dcPlot(does,tc=15*60,dmax=1000)

## End(Not run)
```

`deer` *GPS tracking data of two male deer*

Description

GPS telemetry data for two male deer during a one-week period in March 2005. The two deer form a male bachelor group, making them an interesting case study for studying dynamic interaction patterns. The data are a subset of the data used as a case study in Long *et al.* (2014).

Format

An `ltraj` object with two bursts, representing the two different individual deer:

- Deer no. 37 containing 551 fixes.
- Deer no. 38 containing 567 fixes.

Details

The deer data are stored as a single `'ltraj'` object; two bursts contain the fixes for two individuals (deer37 and deer 38). GPS fixes were attempted at a regular sampling frequency of 15 minutes. For more information on these data how the deer data was collected or for citation please see the papers Webb *et al.* (2009, 2010).

References

Long, J.A., Nelson, T.A., Webb, S.L., Gee, K.L. (2014) A critical examination of indices of dynamic interaction for wildlife telemetry studies. *Journal of Animal Ecology*, **83**: 1216-1233.

Webb, S.L., Gee, K.L., Demarais, S., Strickland, B.K., DeYoung, R.W. (2009) Efficacy of a 15-strand high-tensile electric fence to control white-tailed deer movements. *Wildlife Biology in Practice*, **5**, 45-57.

Webb, S.L., Gee, K.L., Strickland, B.K., Demarais, S., DeYoung, R.W. (2010) Measuring fine-scale white-tailed deer movements and environmental influences using GPS collars. *International Journal of Ecology*, **2010**, 1-12.

Examples

```
data(deer)
deer37 <- deer[1]
deer38 <- deer[2]
plot(deer37)
plot(deer38)
```

DI Dynamic interaction index

Description

The function `DI` measures dynamic interaction between two moving objects. It calculates the local level di statistic for movement displacement, direction, and overall. DI can compute time- and/or distance-based weighting schemes following Long and Nelson (2013).

Usage

```
DI(traj, traj2, tc = 0, local = FALSE, rand = 0, alpha = 1)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`local`	logical value indicating whether a dataframe (`local` = `TRUE`) containing the DI index for each simultaneous fix should be returned (with a local permutation test), or (if `local` = `FALSE` - the default) the global index along with associated global permutation test.
`rand`	number of permutations to use in the local permutation test.
`alpha`	value for the α parameter in the formula for di_d (default = 1).

Details

This function can be used for calculating the dynamic interaction (DI) statistic as described in Long and Nelson (2013). The DI statistic can be used to measure the local level of dynamic interaction between two moving objects. Specifically, it measures dynamic interaction in movement direction and displacement.

Value

If `local=FALSE` (the default) DI returns the numeric value of the DI index (along with DI_{theta} and DI_d), and the associated p-value from a permutation test (see `IAB`). If `local=TRUE` DI returns a large dataframe that contains the localized `di` values as a column (see Long and Nelson 2013). The columns for `di`, `di.theta`, and `di.d` represent dynamic interaction overall, in direction (azimuth), and in displacement, respectively for each segment. A localized p-value for a one sided test for positive interaction (and z-score) is computed based on `rand` permutations of the segments. The row.name columns can be used to match the simultaneous segments to the original trajectory (see `IAB`).

References

Long, J.A., Nelson, T.A. 2013. Measuring dynamic interaction in movement data. *Transactions in GIS*. 17(1): 62-77.

See Also

GetSimultaneous, Cr, IAB

Examples

```
## Not run:
data(deer)
#tc = 7.5 minutes
DI(deer, tc = 7.5*60)
df <- DI(deer, tc = 7.5*60, local = TRUE)

## End(Not run)
```

`does` *GPS tracking data of female white-tailed deer*

Description

GPS telemetry data for 8 does during month of May in 2011.

Format

An `ltraj` object with where bursts represent different individual deer.

Details

The doe data are stored as a single `'ltraj'` object; each burst represents an individual.

Examples

```
data(does)
plot(does)
```

Don *Doncaster's measure of dynamic interaction*

Description

The function `Don` measures the dynamic interaction between two moving objects following the methods outlined by Doncaster (1990).

Usage

```
Don(traj, traj2, tc = 0, dc = 0, plot = TRUE)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`dc`	distance tolerance limit (in appropriate units) for defining when two fixes are spatially together.
`plot`	logical, whether or not to plot the Doncaster plot. Default = TRUE.

Details

This function can be used to compute the Doncaster (1990) methods for measuring dynamic interaction between two objects. The Doncaster method tests the proportion of simultaneous fixes that are below `dc` against that which would be expected based on the distribution of distances between all fixes.

Value

A data.frame is returned that contains the values for the contingency table of simultaneous fixes (paired) and non-paired fixes below and above `dc`, along with the associated *p*-value from the Chi-squared test.This function can optionally return a plot, for distance values ranging from 0 to the maximum distance separating two fixes, of the observed proportion of simultaneous fixes below each distance value (for each pair). The expected values based on all fixes are also included.

References

Doncaster, C.P. (1992) Non-parametric estimates of interaction from radio-tracking data. *Journal of Theoretical Biology*, **143**: 431-443.

See Also

GetSimultaneous

Examples

```
data(deer)
#tc = 7.5 minutes, dc = 50 meters
Don(deer, tc = 7.5*60, dc = 50)
```

`GetSimultaneous` *Identify simultaneous fixes between trajectories*

Description

The function `GetSimultaneous` identifies and extracts simultaneous fixes, within a given tolerance limit, between two movement trajectories.

Usage

```
GetSimultaneous(traj1, traj2, tc = 0)
```

Arguments

`traj1`	an object of the class `move2` which contains the time-stamped movement fixes of EXACTLY 1 individual. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	same as `traj1`.
`tc`	time threshold for determining simultaneous fixes. For simplicity, `tc` is always taken in seconds.

Details

This function is used to determine the simultaneous fixes between two movement datasets facilitating further analysis.

Value

A `move2` object containing two individuals, representing the two original `move2` objects, each containing only those fixes that are deemed simultaneous.

See Also

GetTO

Examples

```
library(move2)
data(deer)
deer37 <- deer[mt_track_id(deer) == '37',]
deer38 <- deer[mt_track_id(deer) == '38',]
deer_sim <- GetSimultaneous(deer37, deer38, tc = 7.5*60)
table(deer$id)
table(deer_sim$id)
```

GetTO *Get period where two tracks overlap*

Description

The function `GetTO` identifies and extracts fixes of a tracking dataset that overlap in time with all other trajectories.

Usage

```
GetTO(traj, tb = 0)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of >= 1 individual. For more information on objects of this type see `help(mt_as_move2)`.
`tb`	(optional) time threshold (i.e., time buffer) for considering if fixes are in the overlap period (in seconds).

Details

This function is used to determine the fixes that overlap in time between two trajectories.

Value

A single `move2` object containing the fixes from traj that temporally overlap. If more than 2 individuals it will take the maximum of the earliest start-time from all individuals and minimum of the latest end-time of all individuals.

See Also

checkTO

Examples

```
data(deer)
deer_to <- GetTO(deer)
```

`HAI` *Half-weight Association Index*

Description

This function computes the Half-weight Association Index for examining the presence of dynamic interaction in wildlife telemetry studies. This implementation follows that outlined in the paper Atwood and Weeks (2003).

Usage

```
HAI(traj, traj2, hr = NULL, tc = 0, dc = 50)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`hr`	(optional)spatial polygon `sf` object associated with the home range (or some other form of) spatial range estimate for each individual in `traj`. The hr polygon should have a corresponding ID column with the same column name as in `traj`. If NULL (the default) the MCP home range estimate will be used for each individual.
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`dc`	distance tolerance limit (in appropriate units) for defining when two fixes are spatially together.

Details

This function can be used to test for the presence of dynamic interaction within the shared area (often termed the overlap zone) of the two animals home ranges. Specifically, HAI is calculated in identical fashion to that for `Ca`, but considers only those fixes in the shared area. Typically, the overlap zone (OZ) is easily obtained by taking the spatial intersection of two polygon home ranges.

Value

This function returns the numeric value of the HAI statistic. Values near 1 indicate attraction within the shared home range area, while values near 0 indicate avoidance within this shared area.

References

Atwood, T.C. and Weeks Jr., H.P. (2003) Spatial home-range overlap and temporal interaction in eastern coyotes: The influence of pair types and fragmentation. *Canadian Journal of Zoology*, **81**: 1589-1597.

See Also

GetSimultaneous, Ca

Examples

```
## Not run:
data(deer)

#uses as a default minimum convex polygon for home range...
#tc = 7.5 minutes, dc = 50 meters
HAI(deer, tc=7.5*60, dc=50)

## End(Not run)
```

`IAB` *Benhamou's IAB Index*

Description

The function `IAB` computes the IAB index following the methods described in the paper by Benhamou et al. (2014). It facilitates global analysis, with the significance testing procedure described in the paper, but also a local level output, to explore the IAB statistic through time.

Usage

```
IAB(traj, traj2, tc = 0, dc = 0, local = FALSE, rand = 99)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`dc`	critical distance where the IAB function will show maximum slope – see Benhamou et al. (2014) for more advice on selecting this parameter.
`local`	logical value indicating whether a dataframe (`local` = `TRUE`) containing the IAB index for each simultaneous fix should be returned (with a local permutation test), or (if `local` = `FALSE` - the default) the global index along with associated global permutation test.
`rand`	number of permutations to use in the local permutation test.

Details

The function `IAB` can be used to test for direct interaction in wildlife telemetry data and affords a novel significance testing procedure that takes into account the serially correlated structure of telemetry data. Specifically, it computes an index analogous to the Bhattacharyya coefficient between the potential influence domains of two animals. Like the other indices, IAB is dependent on the selection of an appropriate value for `dc` (which is termed Δ in the article). The `dc` parameter here is not a threshold distance, but rather the distance at which the function shows maximum slope (see Benhamou et al. 2014).

The significance testing procedure uses a wrapped shifting method in order to maintain the serially correlated structure of the data. At each shift, a sample value of IAB (termed MAB) is computed in order to generate a distribution of values to test against (for more information see Benhamou et al. 2014). Here a local version of this statistical testing procedure is implemented by taking `rand` samples of the $(n^2 - n$ permutations of unpaired fixes. The p-values are computed following Benhamou et al. (2014), z-scores are calculated based on the mean and standard deviation of this hypothetical distribution.

Value

If `local=FALSE` (the default) IAB returns a dataframe with the values of the IAB index and the associated p-values for one-sided tests for attraction or avoidance. If `local=TRUE` IAB returns a dataframe (containing the date/times of *all* simultaneous fixes (NOTE: times are associated with traj1), along with the distance between fixes at each time , and the IAB index value for each simultaneous fix. A localized p-value (.pa signifies the test for attraction and pb the test for avoidance) and z-score is computed based on `rand` permutations of the fixes. The row.name columns can be used to match the simultaneous fixes to the original trajectory.

References

Benhamou, S., Valeix, M., Chamaille-Jammes, S., Macdonald, D., Loveridge, A.J. (2014) Movement-based analysis of interactions in African lions. *Animal Behaviour*, **90**: 171-180.

See Also

GetSimultaneous, DI, Prox

Examples

```
data(deer)
#tc = 7.5 minutes, dc = 50 meters
IAB(deer, tc=7.5*60, dc=50)
df <- IAB(deer, tc=7.5*60, dc=50, local=TRUE)
```

`Lixn` *Minta's Spatial-temporal interaction statistics*

Description

The function `Lixn` measures dynamic interaction between two animals following the methods outlined by Minta (1992).

Usage

```
Lixn(traj, traj2, method = "spatial", tc = 0, hr = NULL, OZ = NULL)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`method`	method for computing the marginal distribution from which expected values are computed. If `method = "spatial"`, the marginal values are calculated based on areas of the shared and unshared portions of the home ranges. If `method = "frequency"`, the marginal values are calculated based on the number of all fixes within the shared and unshared portions of the home ranges – see Details.
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`hr`	(optional) spatial polygon `sf` object associated with the home range (or some other form of) spatial range estimate for each individual in `traj`. The hr polygon should have a corresponding ID column with the same column name as in `traj`. If NULL (the default) the MCP home range estimate will be used for each individual.
`OZ`	(– required if method = 'frequency') A `sf` object representing the shared area polygon associated with spatial use overlap each pair of individuals in `traj`. Must be a `sf` polygon object and contain two columns id1 and id2 indicating the polygon associated with each pair.

Details

The function `Lixn` can be used to calculate the Minta (1992) measures of dynamic interaction between two animals. The Minta statistic tests how the two animals simultaneously utilize an area shared between the two individuals. Three coefficients are produced L_{AA}, L_{BB}, and L_{ixn}. Each of these statistics are based on a contingency table that compares the observed frequency of those fixes that are simultaneous and within/outside the shared area to expectations based on area overlap proportions (if `method="spatial"`) or expectations derived from all fixes (if `method="frequency"`) – see Minta (1992) for more details. A Chi-squared statistic can then be used to examine the significance between the observed and expected use of the shared area.

Minta (1992) suggests the following interpretations of the coefficients. When L_{AA} is near 0, the first animal's use of the shared area is random (or as expected). When $L_{AA} > 0$ it signifies spatial attraction to the shared area, or greater than expected use. When $L_{AA} < 0$ it signifies spatial avoidance of the shared area, or less than expected use. Interpretation of L_{BB} is the same as for L_{AA} with respect to the second animal. L_{ixn} tells us far more about the nature of the interaction between the two individuals. As L_{ixn} nears 0, both animals use the shared area randomly, with regards to the other animal. If $L_{ixn} > 0$ the animals use the shared area more *simultaneously*, whereas if $L_{ixn} < 0$ it is an indication of *solitary* use, or avoidance. This is why L_{ixn} is termed the temporal interaction coefficient. A Chi-squared test can be used to identify the significance of the L_{AA}, L_{BB}, and L_{ixn} values.

NOTEs:
1. With modern telemetry datasets, where home ranges are readily estimated, choosing `method = 'spatial'` is most appropriate. If parmater `hr` is not specified, the code uses the minimum convex hull method to calculate individual home ranges.
2. When the home ranges do not overlap the Lixn statistic is not defined and the function returns a string of NA's.
3. When one home range completely encloses another the Lixn statistic is not defined and the function returns a string of NA's and `'ContainsB'` (or `'ContainsB'`) under the p.IXN result.
4. Further to points 2 and 3, the Lixn statistic is not appropriate in situations where the overlap area is either very large or very small relative to either home range (i.e., a situation with almost complete enclosure or virtually no overlap). The example data (deer) is an exampl of a near complete enclosure. Thus, it is advised that `Lixn` be used only in situations where there are suitable marginal areas for areaA, areaB, and areaAB – see Minta (1992).

Value

This function returns a data.frame with values representing the calculated statistical values and associated *p*-values from the Chi-squared test for each dyad.

- pTable – contingency table showing marginal probabilities of expected use based on the selection of the `method` parameter.
- nTable – contingency table showing observed frequency of use of the shared area based on simultaneous fixes.
- oTable – the odds for each cell in the contingency table.
- Laa – the calculated value of the L_{AA} statistic
- p.AA – the associated *p*-value
- Lbb – the calculated value of the L_{BB} statistic
- p.BB – the associated *p*-value
- Lixn – the calculated value of the L_{ixn} statistic
- p.IXN – the associated *p*-value

References

Minta, S.C. (1992) Tests of spatial and temporal interaction among animals. *Ecological Applications*, **2**: 178-188

See Also

GetSimultaneous

Examples

```
## Not run:
data(deer)
#tc = 7.5 minutes, dc = 50 meters
Lixn(deer,  method='spatial', tc=7.5*60)

#use internal buffer 500m of MCP for demonstration of frequency method
# NOTE: This is just an example, this is not an appropriate way to define overlap zone.
idcol <- mt_track_id_column(deer)
deercore <- deer |>
  st_union() |>
  st_convex_hull() |>
  st_buffer(-500)
Lixn(deer,  method='frequency', tc=7.5*60, OZ=deercore)

## End(Not run)
```

`ltraj_move2` *Convert ltraj to move2 object*

Description

The function `ltraj_move2` is a simple function for quickly converting `ltraj` to `move2` objects.

Usage

```
ltraj_move2(ltraj)
```

Arguments

`ltraj`	an object of the class `ltraj` which contains the time-stamped movement fixes of the object. For more information on objects of this type see `help(ltraj)`.

Details

The function `ltraj_move2` can be used to convert an `ltraj` object into an `move2` object.

Value

A `move2` object.

See Also

move2_ltraj

Examples

```
data(deer)
deer_ltraj <- move2_ltraj(deer)
deer_move <- ltraj_move2(deer_ltraj)
```

`move2_ltraj` *Convert move2 to ltraj object*

Description

The function `move2_ltraj` is a simple function for quickly converting `move2` to `ltraj` objects.

Usage

```
move2_ltraj(traj)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of the object. For more information on objects of this type see `help(move2)`.

Details

The function `move2_ltraj` can be used to convert a `move2` object into a `ltraj` object.

Value

A `ltraj` object.

See Also

ltraj_move2

Examples

```
data(deer)
deer_ltraj <- move2_ltraj(deer)
```

`Prox` *Proximity Index*

Description

The function `Prox` simply computes the proportion of (simultaneous) fixes that are proximal, based on some spatial threshold – `dc` (Bertrand et al. 1996). It also facilitates local-level proximity analysis

Usage

```
Prox(traj, traj2, tc = 0, dc = 50, local = FALSE)
```

Arguments

`traj`	an object of the class `move2` which contains the time-stamped movement fixes of at least two individuals. For more information on objects of this type see `help(mt_as_move2)`.
`traj2`	(optional) same as traj, but for the second group of individuals. See `checkTO`
`tc`	time threshold for determining simultaneous fixes – see function: `GetSimultaneous`.
`dc`	distance tolerance limit (in appropriate units) for defining when two fixes are spatially together.
`local`	logical value indicating. When local = FALSE (the default) prox returns a data.frame with the global proximity ratio (proportion of all fixes below dc and tc) for each pair of individuals. When local = TRUE, prox returns the input move2 object with the distance to the most proximal fix, and the number of fixes that are considered proximal for each fix in the dataset.

Details

The function `Prox` can be used to test for the presence of attraction (via proximity) in wildlife telemetry data. Prox is simply the proportion of simultaneous fixes within the threshold distance – `dc`. The local output (dataframe) can be useful for examining variation in proximity through time.

Value

If `local=FALSE` (the default) Prox returns the numeric value of the Prox index for each pair of individuals. If `local=TRUE` Prox returns a `move2` containing the original trajectory (or both trajectories) with three additional columns prox (the distance to the nearest proximal fix), prox.id (the id of the nearest proximal fix) and prox.n (the number of individuals with proximal fixes)

References

Bertrand, M.R., DeNicola, A.J., Beissinger, S.R, Swihart, R.K. (1996) Effects of parturition on home ranges and social affiliations of female white-tailed deer. *Journal of Wildlife Management*, **60**: 899-909.

See Also

GetSimultaneous, contacts

Examples

```
data(deer)
#tc = 7.5 minutes, dc = 50 meters
Prox(deer, tc=7.5*60, dc=50)
deer <- Prox(deer, tc=7.5*60, dc=50, local=TRUE)
```

Index

∗ **contacts**
  conDisplacement, 4
  conPhase, 6
  conProcess, 7
  conTimelag, 8
∗ **datasets**
  deer, 13
  does, 15
∗ **indices**
  Ca, 2
  Cr, 9
  Cs, 10
  DI, 14
  Don, 16
  HAI, 19
  IAB, 20
  Lixn, 22
  Prox, 26
∗ **processing**
  checkTO, 3
  GetSimultaneous, 17
  GetTO, 18
  ltraj_move2, 24
  move2_ltraj, 25

Ca, 2
checkTO, 3
conDisplacement, 4
conPhase, 6
conProcess, 7
conTimelag, 8
Cr, 9
Cs, 10

dcPlot, 12
deer, 13
DI, 14
does, 15
Don, 16
GetSimultaneous, 17
GetTO, 18

HAI, 19

IAB, 20

Lixn, 22
ltraj_move2, 24

move2_ltraj, 25

Prox, 26