Package 'ecotraj'

June 8, 2021

Type Package

Title Ecological Trajectory Analysis

Version 0.0.1

Date 2021-05-30

Description Assists ecologists in the analysis of temporal changes of ecosystems, defined as trajectories on a chosen multivariate space, by providing a set of trajectory metrics and visual representations (see De Caceres et al. (2019) <doi:10.1002/ecm.1350> and Sturbois et al. (2021) <doi:10.1016/j.ecolmodel.2020.109400>). Includes functions to estimate metrics for individual trajectories (length, directionality, angles, ...) as well as metrics to relate pairs of trajectories (dissimilarity and convergence).

Depends R (>= 3.4.0), Rcpp (>= 0.12.12)

Imports Kendall, circular, MASS

LinkingTo Rcpp

License GPL (>= 2)

URL https://emf-creaf.github.io/ecotraj/

LazyLoad yes

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.1.1

Suggests vegclust, knitr, rmarkdown, RColorBrewer, smacof, vegan

VignetteBuilder utils, knitr

Author Miquel De Cáceres [aut, cre],
Anthony Sturbois [aut]

Maintainer Miquel De Cáceres <miquelcaceres@gmail.com>

Repository CRAN

Date/Publication 2021-06-08 07:00:05 UTC

R topics documented:

---

avoca *Avoca permanent plot dataset*

Description

Example dataset with data from 8 permanent forest plots located on slopes of a valley in the New Zealand Alps. The study area is mountainous and centered on the Craigieburn Range (Southern Alps), South Island, New Zealand. Forests plots are almost monospecific, being the mountain beech (Fuscospora cliffortioides) the main dominant tree species. Previously forests consisted of largely mature stands, but some of them were affected by different disturbances during the sampling period (1972-2009) which includes 9 surveys.

Details

- avoca_strat An object of class `stratifiedvegdata` (see function `stratifyvegdata` from package 'vegclust') with structural and compositional data.
- avoca_sites A vector identifying sampled sites of each element in `avoca_strat`.
- avoca_surveys A vector identifying surveys of each element in `avoca_strat`.

Author(s)

New Zealand National Vegetation Survey (NVS) Databank (https://nvs.landcareresearch.co.nz/).

References

Allen, R. B., P. J. Bellingham, and S. K. Wiser. 1999. Immediate damage by an earthquake to a temperate montane forest. Ecology 80:708–714.

Harcombe, P. A., R. B. Allen, J. A. Wardle, and K. H. Platt. 1998. Spatial and temporal patterns in stand structure, biomass, growth and mortality in a monospecific Nothofagus solandri var. cliffortioides (Hook. f.) Poole forest in New Zealand. Journal of Sustainable Forestry 6:313–343.

Hurst, J. M., R. B. Allen, D. A. Coomes, and R. P. Duncan. 2011. Size-specific tree mortality varies with neighbourhood crowding and disturbance in a montane Nothofagus forest. PLoS ONE 6.

`trajectorymetrics` *Metrics for Ecological Trajectory Analysis*

Description

Ecological Trajectory Analysis (ETA) is a framework to analyze dynamics of ecosystems described as trajectories in a chosen space of multivariate resemblance (De Cáceres et al. 2019). ETA takes trajectories as objects to be analyzed and compared geometrically.

Usage

```
segmentDistances(
  d,
  sites,
  surveys = NULL,
  distance.type = "directed-segment",
  add = TRUE,
  verbose = FALSE
)

trajectoryDistances(
  d,
  sites,
  surveys = NULL,
  distance.type = "DSPD",
  symmetrization = "mean",
  add = TRUE,
  verbose = FALSE
)

trajectoryLengths(
  d,
  sites,
  surveys = NULL,
  relativeToInitial = FALSE,
  all = FALSE,
  verbose = FALSE
)

trajectoryLengths2D(
  xy,
  sites,
  surveys,
  relativeToInitial = FALSE,
  all = FALSE,
  verbose = FALSE
)
```

```
trajectoryAngles(
  d,
  sites,
  surveys = NULL,
  all = FALSE,
  relativeToInitial = FALSE,
  stats = TRUE,
  add = TRUE,
  verbose = FALSE
)

trajectoryAngles2D(
  xy,
  sites,
  surveys,
  relativeToInitial = FALSE,
  betweenSegments = TRUE
)

trajectoryProjection(d, target, trajectory, tol = 1e-06, add = TRUE)

trajectoryConvergence(
  d,
  sites,
  surveys = NULL,
  symmetric = FALSE,
  add = TRUE,
  verbose = FALSE
)

trajectoryDirectionality(d, sites, surveys = NULL, add = TRUE, verbose = FALSE)
```

Arguments

d	A symmetric matrix or an object of class dist containing the distance values between pairs of ecosystem states (see details).
sites	A vector indicating the site corresponding to each ecosystem state.
surveys	A vector indicating the survey corresponding to each ecosystem state (only necessary when surveys are not in order).
distance.type	The type of distance index to be calculated (Besse et al. 2016; De Cáceres et al. submitted). For segmentDistances the available indices are: • Hausdorff: Hausdorff distance between two segments. • directed-segment: Directed segment distance (default). • PPA: Perpendicular-parallel-angle distance. whereas for trajectoryDistances the available indices are: • Hausdorff: Hausdorff distance between two trajectories.

- SPD: Segment path distance.
- DSPD: Directed segment path distance (default).

`add` Flag to indicate that constant values should be added (local transformation) to correct triplets of distance values that do not fulfill the triangle inequality.

`verbose` Provides console output informing about process (useful for large dataset).

`symmetrization` Function used to obtain a symmetric distance, so that DSPD(T1,T2) = DSPD(T2,T1) (e.g., `mean` or `min`). If `symmetrization = NULL` then the symmetrization is not conducted and the output dissimilarity matrix is not symmetric.

`relativeToInitial` Flag to indicate that lengths or angles should be calculated with respect to initial survey.

`all` A flag to indicate that angles are desired for all triangles (i.e. all pairs of segments) in the trajectory. If FALSE, angles are calculated for consecutive segments only.

`xy` Matrix with 2D coordinates in a Cartesian space (typically an ordination of ecosystem states).

`stats` A flag to indicate that circular statistics are desired (mean, standard deviation and mean resultant length, i.e. rho)

`betweenSegments` Flag to indicate that angles should be calculated between trajectory segments or with respect to X axis.

`target` An integer vector of the ecosystem states to be projected.

`trajectory` An integer vector of the trajectory onto which target states are to be projected.

`tol` Numerical tolerance value to determine that projection of a point lies within the trajectory.

`symmetric` A logical flag to indicate a symmetric convergence comparison of trajectories.

Details

Given a distance matrix between ecosystem states, the set of functions that provide ETA metrics are:

- Functions `segmentDistances` and `trajectoryDistances` calculate the distance between pairs of directed segments and ecosystem trajectories, respectively.
- Function `trajectoryLengths` calculates lengths of directed segments and total path lengths of trajectories.
- Function `trajectoryLengths2D` calculates lengths of directed segments and total path lengths of trajectories from 2D coordinates given as input.
- Function `trajectoryAngles` calculates the angle between consecutive pairs of directed segments or between segments of ordered triplets of points.
- Function `trajectoryAngles2D` calculates the angle between consecutive pairs of directed segments or between segments of ordered triplets of points.
- Function `trajectoryProjection` projects a set of target points onto a specified trajectory and returns the distance to the trajectory (i.e. rejection) and the relative position of the projection point within the trajectory.

- Function `trajectoryConvergence` performs the Mann-Kendall trend test on the distances between trajectories (symmetric test) or the distance between points of one trajectory to the other.
- Function `trajectoryDirectionality` returns (for each trajectory) a statistic that measures directionality of the whole trajectory.

Details of calculations are given in De Cáceres et al (2019). The input distance matrix `d` should ideally be metric. That is, all subsets of distance triplets should fulfill the triangle inequality (see utility function `is.metric`). All ETA functions that require metricity include a parameter 'add', which by default is TRUE, meaning that whenever the triangle inequality is broken the minimum constant required to fulfill it is added to the three distances. If such local (an hence, inconsistent across triplets) corrections are not desired, users should find another way modify `d` to achieve metricity, such as PCoA, metric MDS or non-metric MDS (see vignette 'Introduction to Ecological Trajectory Analysis'). If parameter 'add' is set to FALSE and problems of triangle inequality exist, ETA functions may provide missing values in some cases where they should not.

The resemblance between trajectories is done by adapting concepts and procedures used for the analysis of trajectories in space (i.e. movement data) (Besse et al. 2016).

Function `trajectoryAngles` calculates angles between consecutive segments in degrees. For each pair of segments, the angle between the two is defined on the plane that contains the two segments, and measures the change in direction (in degrees) from one segment to the other. Angles are always positive, with zero values indicating segments that are in a straight line, and values equal to 180 degrees for segments that are in opposite directions. If `all = TRUE` angles are calculated between the segments corresponding to all ordered triplets. Alternatively, if `relativeToInitial = TRUE` angles are calculated for each segment with respect to the initial survey.

Function `trajectoryAngles2D` calculates angles between consecutive segments in degrees from 2D coordinates given as input. For each pair of segments, the angle between the two is defined on the plane that contains the two segments, and measures the change in direction (in degrees) from one segment to the other. Angles are always positive (O to 360), with zero values indicating segments that are in a straight line, and values equal to 180 degrees for segments that are in opposite directions. If `all = TRUE` angles are calculated between the segments corresponding to all ordered triplets. Alternatively, if `relativeToInitial = TRUE` angles are calculated for each segment with respect to the initial survey. If `betweenSegments = TRUE` angles are calculated between segments of trajectory, otherwise, If `betweenSegments = FALSE`, angles are calculated considering Y axis as the North (0°).

Value

Function `trajectoryDistances` returns an object of class `dist` containing the distances between trajectories (if `symmetrization = NULL` then the object returned is of class `matrix`).

Function `trajectorySegments` returns a list with the following elements:

- `Dseg`: Distance matrix between segments.
- `Dini`: Distance matrix between initial points of segments.
- `Dfin`: Distance matrix between final points of segments.
- `Dinifin`: Distance matrix between initial points of one segment and the final point of the other.

- `Dfinini`: Distance matrix between final points of one segment and the initial point of the other.

Function `trajectoryLengths` returns a data frame with the length of each segment on each trajectory and the total length of all trajectories. If `relativeToInitial = TRUE` lengths are calculated between the initial survey and all the other surveys. If `all = TRUE` lengths are calculated for all segments.

Function `trajectoryLengths2D` returns a data frame with the length of each segment on each trajectory and the total length of all trajectories. If `relativeToInitial = TRUE` lengths are calculated between the initial survey and all the other surveys. If `all = TRUE` lengths are calculated for all segments.

Function `trajectoryAngles` returns a data frame with angle values on each trajectory. If `stats=TRUE`, then the mean, standard deviation and mean resultant length of those angles are also returned.

Function `trajectoryAngles2D` returns a data frame with angle values on each trajectory. If `betweenSegments=TRUE`, then angles are calculated between trajectory segments, alternatively, If `betweenSegments=FALSE`, angles are calculated considering Y axis as the North (0°).

Function `trajectoryProjection` returns a data frame with the following columns:

- `distanceToTrajectory`: Distances to the trajectory, i.e. rejection (`NA` for target points whose projection is outside the trajectory).
- `segment`: Segment that includes the projected point (`NA` for target points whose projection is outside the trajectory).
- `relativePosition`: Relative position of the projected point within the trajectory, i.e. values from 0 to 1 with 0 representing the start of the trajectory and 1 representing the end (`NA` for target points whose projection is outside the trajectory).

Function `trajectoryConvergence` returns a list with two elements:

- `tau`: A matrix with the statistic (Mann-Kendall's tau) of the convergence/divergence test between trajectories. If `symmetric=TRUE` then the matrix is square. Otherwise the statistic of the test of the row trajectory approaching the column trajectory.
- `p.value`: A matrix with the p-value of the convergence/divergence test between trajectories. If `symmetric=TRUE` then the matrix is square. Otherwise the p-value indicates the test of the row trajectory approaching the column trajectory.

Function `trajectoryDirectionality` returns a vector with directionality values (one per trajectory).

Author(s)

Miquel De Cáceres, CREAF

Anthony Sturbois, Vivarmor nature, Réserve Naturelle nationale de la Baie de Saint-Brieuc

References

Besse, P., Guillouet, B., Loubes, J.-M. & François, R. (2016). Review and perspective for distance based trajectory clustering. IEEE Trans. Intell. Transp. Syst., 17, 3306–3317.

De Cáceres M, Coll L, Legendre P, Allen RB, Wiser SK, Fortin MJ, Condit R & Hubbell S. (2019). Trajectory analysis in community ecology. Ecological Monographs.

See Also

trajectoryplots, trajectoryutils

Examples

```
#Description of sites and surveys
sites = c(1,1,1,2,2,2)
surveys=c(1,2,3,1,2,3)

#Raw data table
xy<-matrix(0, nrow=6, ncol=2)
xy[2,2]<-1
xy[3,2]<-2
xy[4:6,1] <- 0.5
xy[4:6,2] <- xy[1:3,2]
xy[6,1]<-1

#Draw trajectories
trajectoryPlot(xy, sites, surveys,
               traj.colors = c("black","red"), lwd = 2)

#Distance matrix
d = dist(xy)
d

trajectoryLengths(d, sites, surveys)
trajectoryLengths2D(xy, sites, surveys)
trajectoryAngles(d, sites, surveys)
trajectoryAngles2D(xy, sites, surveys, betweenSegments = TRUE)
trajectoryAngles2D(xy, sites, surveys, betweenSegments = FALSE)
segmentDistances(d, sites, surveys)$Dseg
trajectoryDistances(d, sites, surveys, distance.type = "Hausdorff")
trajectoryDistances(d, sites, surveys, distance.type = "DSPD")


#Should give the same results if surveys are not in order
#(here we switch surveys for site 2)
temp = xy[5,]
xy[5,] = xy[6,]
xy[6,] = temp
surveys[5] = 3
surveys[6] = 2

trajectoryPlot(xy, sites, surveys,
               traj.colors = c("black","red"), lwd = 2)
trajectoryLengths(dist(xy), sites, surveys)
trajectoryLengths2D(xy, sites, surveys)
segmentDistances(dist(xy), sites, surveys)$Dseg
trajectoryAngles(dist(xy), sites, surveys)
trajectoryAngles2D(xy, sites, surveys, betweenSegments = TRUE)
trajectoryAngles2D(xy, sites, surveys, betweenSegments = FALSE)
trajectoryDistances(dist(xy), sites, surveys, distance.type = "Hausdorff")
```

```
trajectoryDistances(dist(xy), sites, surveys, distance.type = "DSPD")
```

`trajectoryplots` *Trajectory plots*

Description

Set of plotting functions for Ecological Trajectory Analysis:

Usage

```
trajectoryPCoA(
  d,
  sites,
  surveys = NULL,
  selection = NULL,
  traj.colors = NULL,
  axes = c(1, 2),
  survey.labels = FALSE,
  ...
)

trajectoryPlot(
  x,
  sites,
  surveys = NULL,
  selection = NULL,
  traj.colors = NULL,
  axes = c(1, 2),
  survey.labels = FALSE,
  ...
)
```

Arguments

`d`	A symmetric `matrix` or an object of class `dist` containing the distance values between pairs of ecosystem states (see details).
`sites`	A vector indicating the site corresponding to each ecosystem state.
`surveys`	A vector indicating the survey corresponding to each ecosystem state (only necessary when surveys are not in order).
`selection`	A character vector of sites, a numeric vector of site indices or logical vector of the same length as `sites`, indicating a subset of site trajectories to be selected.
`traj.colors`	A vector of colors (one per site). If `selection != NULL` the length of the color vector should be equal to the number of sites selected.
`axes`	The pair of principal coordinates to be plotted.

survey.labels	A boolean flag to indicate whether surveys should be plotted as text next to arrow endpoints
...	Additional parameters for function arrows.
x	A data.frame or matrix where rows are ecosystem states and columns are coordinates in an arbitrary space

Details

- Function trajectoryPCoA performs principal coordinates analysis (cmdscale) and draws trajectories in the ordination scatterplot.
- Function trajectoryPlot Draws trajectories in a scatterplot corresponding to the input coordinates.

Details of calculations are given in De Cáceres et al (2019). The input distance matrix d should ideally be metric. That is, all subsets of distance triplets should fulfill the triangle inequality (see function is.metric). All CTA functions that require metricity include a parameter 'add', which by default is TRUE, meaning that whenever the triangle inequality is broken the minimum constant required to fulfill it is added to the three distances. If such local (an hence, inconsistent across triplets) corrections are not desired, users should find another way modify d to achieve metricity, such as PCoA, metric MDS or non-metric MDS (see CTA vignette). If parameter 'add' is set to FALSE and problems of triangle inequality exist, CTA functions may provide missing values in some cases where they should not.

The resemblance between trajectories is done by adapting concepts and procedures used for the analysis of trajectories in space (i.e. movement data) (Besse et al. 2016).

Value

Function trajectoryPCoA returns the result of calling cmdscale.

Author(s)

Miquel De Cáceres, CREAF

Anthony Sturbois, Vivarmor nature, Réserve Naturelle nationale de la Baie de Saint-Brieuc

References

Besse, P., Guillouet, B., Loubes, J.-M. & François, R. (2016). Review and perspective for distance based trajectory clustering. IEEE Trans. Intell. Transp. Syst., 17, 3306–3317.

De Cáceres M, Coll L, Legendre P, Allen RB, Wiser SK, Fortin MJ, Condit R & Hubbell S. (2019). Trajectory analysis in community ecology. Ecological Monographs.

Anderson (2017). Permutational Multivariate Analysis of Variance (PERMANOVA). Wiley StatsRef: Statistics Reference Online. 1-15. Article ID: stat07841.

See Also

trajectorymetrics, trajectoryutils, cmdscale

Examples

```
#Description of sites and surveys
sites = c(1,1,1,2,2,2)
surveys=c(1,2,3,1,2,3)

#Raw data table
xy<-matrix(0, nrow=6, ncol=2)
xy[2,2]<-1
xy[3,2]<-2
xy[4:6,1] <- 0.5
xy[4:6,2] <- xy[1:3,2]
xy[6,1]<-1

#Draw trajectories
trajectoryPlot(xy, sites, surveys,
               traj.colors = c("black","red"), lwd = 2)


#Should give the same results if surveys are not in order
#(here we switch surveys for site 2)
temp = xy[5,]
xy[5,] = xy[6,]
xy[6,] = temp
surveys[5] = 3
surveys[6] = 2

trajectoryPlot(xy, sites, surveys,
               traj.colors = c("black","red"), lwd = 2)
```

`trajectoryutils` *Utility functions for Ecological Trajectory Analysis*

Description

The set following set of utility functions are provided:

- Function `trajectorySelection` allows selecting the submatrix of distances corresponding to a given subset of trajectories.
- Function `centerTrajectories` shifts all trajectories to the center of the compositional space and returns a modified distance matrix.
- Function `is.metric` checks whether the input dissimilarity matrix is metric (i.e. all triplets fulfill the triangle inequality).

Usage

```
trajectorySelection(d, sites, selection)
```

```
centerTrajectories(d, sites, verbose = FALSE)

is.metric(d, tol = 1e-04)
```

Arguments

d	A symmetric matrix or an object of class dist containing the distance values between pairs of ecosystem states (see details).
sites	A vector indicating the site corresponding to each ecosystem state.
selection	A character vector of sites, a numeric vector of site indices or logical vector of the same length as sites, indicating a subset of site trajectories to be selected.
verbose	Provides console output informing about process (useful for large dataset).
tol	Tolerance value for metricity

Details

Details of calculations are given in De Cáceres et al (2019). Function centerTrajectories performs centering of trajectories using matrix algebra as explained in Anderson (2017).

Value

Function centerTrajectories and trajectorySelection return an object of class dist.

Author(s)

Miquel De Cáceres, CREAF

References

De Cáceres M, Coll L, Legendre P, Allen RB, Wiser SK, Fortin MJ, Condit R & Hubbell S. (2019). Trajectory analysis in community ecology. Ecological Monographs.

Anderson (2017). Permutational Multivariate Analysis of Variance (PERMANOVA). Wiley StatsRef: Statistics Reference Online. 1-15. Article ID: stat07841.

See Also

trajectoryplots trajectorymetrics

Index