Package 'codyn'

August 29, 2016

Title Community Dynamics Metrics

Version 1.1.0

Date 2016-04-26

Description A toolbox of ecological community dynamics metrics that are explicitly temporal. Functions fall into two categories: temporal diversity indices and community stability metrics. The diversity indices are temporal analogs to traditional diversity indices such as richness and rank-abundance curves. Specifically, functions are provided to calculate species turnover, mean rank shifts, and lags in community similarity between time points. The community stability metrics calculate overall stability and patterns of species covariance and synchrony over time.

Depends R (>= 3.2.0)

Imports assertthat, stats, permute

Suggests testthat, knitr, ggplot2, reshape2, devtools, dplyr, rmarkdown

License Apache License (== 2.0)

URL https://github.com/laurenmh/codyn/

BugReports https://github.com/laurenmh/codyn/issues

LazyData true

VignetteBuilder knitr

RoxygenNote 5.0.1

NeedsCompilation no

Author Lauren Hallett [aut],
Sydney K. Jones [aut],
A. Andrew M. MacDonald [aut],
Dan F. B. Flynn [aut],
Peter Slaughter [aut],
Julie Ripplinger [aut],
Scott L. Collins [aut],
Corinna Gries [aut],
Matthew B. Jones [aut, cre]

Maintainer Matthew B. Jones <jones@nceas.ucsb.edu>

Repository CRAN

Date/Publication 2016-04-27 13:30:58

R topics documented:

check_multispp *Utility function to stop calculations if only one species occurs in at least one replicate*

Description

Utility function to stop calculations if only one species occurs in at least one replicate

Usage

```
check_multispp(df, species.var, replicate.var)
```

Arguments

df	A dataframe containing time.var, species.var and abundance.var columns
species.var	The name of the species column from df
replicate.var	The name of the replicate column from df

check_names *check names of data frames*

Description

check names of data frames

Usage

```
check_names(given, data)
```

Arguments

given	Vector of variable names as supplied by user
data	Data frame containing variables

check_numeric *Utility to check for numeric abundance and time variables*

Description

Utility to check for numeric abundance and time variables

Usage

```
check_numeric(df, time.var, abundance.var)
```

Arguments

df	A dataframe containing time.var, species.var, and replicate.var columns
time.var	The name of the time column from df
abundance.var	The name of the replicate column from df

check_single *Utility function to ensure only a single record exists for a given species within one replicate, for one time point.*

Description

Utility function to ensure only a single record exists for a given species within one replicate, for one time point.

Usage

```
check_single(df, time.var, species.var, replicate.var)
```

Arguments

df	A dataframe containing time.var, species.var, and replicate.var columns
time.var	The name of the time column from df
species.var	The name of the species column from df
replicate.var	The name of the replicate column from df

check_single_onerep

Utility function to warn users that either multiple records exist within replicates, or that data may be spanning mutiple replicates but no replicate.var has been specified

Description

Utility function to warn users that either multiple records exist within replicates, or that data may be spanning mutiple replicates but no replicate.var has been specified

Usage

```
check_single_onerep(df, time.var, species.var)
```

Arguments

df	A dataframe containing time.var, species.var and abundance.var columns
time.var	The name of the time column from df
species.var	The name of the species column from df

check_sppvar

Utility function to stop calculations if the species never change in a replicate

Description

Utility function to stop calculations if the species never change in a replicate

Usage

```
check_sppvar(comdat)
```

Arguments

comdat	A community dataframe

codyn *Community Dynamics Metrics*

Description

Temporal diversity indices and community stability metrics for ecologists.

Details

The functions in codyn implement metrics that are explicitly temporal, and include the option to calculate them over multiple replicates. Functions fall into two categories: temporal diversity indices and community stability metrics. The diversity indices in codyn are temporal analogs to traditional diversity indices such as richness and rank-abundance curves. Specifically, codyn includes functions to calculate species turnover, mean rank shifts and lags in community similarity between time points. The community stability metrics in codyn calculate overall stability and patterns of species covariance and synchrony over time. Finally, codyn contains vignettes that describe methods and reproduce figures from published papers to help users contextualize and apply functions to their own data. Work on this package was supported by NSF-ABI grant #1262458

Functions

- `turnover`: Calculates species turnover between time periods
- `rank_shift`: Calculates the mean relative change in species rank abundances
- `rate_change`: Calculates the rate change in a community over time
- `rate_change_interval`: Produces a data frame containing differences in species composition between samples at increasing time intervals
- `community_stability`: Calculates community stability over time
- `variance_ratio`: Computes the ratio of the variance of aggregate species abundances in a community
- `synchrony`: Calculates the degree synchrony in species abundances
- `temporal_torus_translation`: Calculates a test statistic on a null ecological community created via cyclic shifts. `confint` provides mean and confidence intervals of this null distribution

Author(s)

- Lauren Hallett <lauren.m.hallett@gmail.com>
- Sydney K. Jones <syd@sevilleta.unm.edu>
- A. Andrew A. MacDonald <aammacdonald@gmail.com>
- Matthew B. Jones <jones@nceas.ucsb.edu>
- Dan F. B. Flynn <flynn@fas.harvard.edu>
- Peter Slaughter <slaughter@nceas.ucsb.edu>
- Corinna Gries <cgries@wisc.edu>
- Scott L. Collins <scollins@sevilleta.unm.edu>

collins08 *Konza data from Collins et al. 2008*

Description

A dataset of tallgrass prairie plant composition at one annually burned and one unburned site over time at the Konza Prairie LTER, Manhattan Kansas (Collins et al. 2008).

Usage

```
data(collins08)
```

Format

A data frame with 2058 rows and 4 variables

Details

A data frame containing a column for replicate, year, species and abundance:

- replicate: A factor column of spatial replicates with two levels ("annually burned" and "unburned")
- year: An integer column of sampling time points
- species: A factor column of species sampled
- abundance: A numeric column of abundance values

Source

Collins, Scott L., Katharine N. Suding, Elsa E. Cleland, Michael Batty, Steven C. Pennings, Katherine L. Gross, James B. Grace, Laura Gough, Joe E. Fargione, and Christopher M. Clark. (2008) "Rank clocks and plant community dynamics." Ecology 89, no. 12: 3534-41.

community_stability *Community Stability*

Description

Calculates the stability of the overall community over time as the temporal mean / temporal standard deviation of aggregate species abundances (Tilman 1999).

Usage

```
community_stability(df, time.var, abundance.var, replicate.var = NA)
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
abundance.var	The name of the abundance column
replicate.var	The name of the optional replicate column

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there should be a single community represented within each time point and replicate.

Value

The community_stability function returns a numeric stability value unless a replication column is specified in the input data frame. If replication is specified, the function returns a data frame with the following attributes:

- stability: A numeric column with the stability values.
- replicate.var: A column that shares the same name and type as the replicate.var column in the input data frame.

References

Tilman, D. "The Ecological Consequences of Changes in Biodiversity: A Search for General Principles." Ecology 80, no. 5 (July 1999): 1455-74. doi:10.1890/0012-9658(1999)080[1455:TECOCI]2.0.CO;2.

Examples

```
data(knz_001d)
community_stability(knz_001d[knz_001d$subplot=="A_1",],
                    time.var = "year",
                    abundance.var = "abundance") # for one subplot
community_stability(knz_001d,
                    time.var = "year",
                    abundance.var = "abundance",
                    replicate.var = "subplot") # across all subplots
```

confint.cyclic_shift *Confidence Intervals from a Cyclic Shift Permutation*

Description

Calculates confidence intervals for the S3 object produced by `cyclic_shift`

Usage

```
## S3 method for class 'cyclic_shift'
confint(object, parm = "out", level = 0.95, ...)
```

Arguments

`object`	An object of class `cyclic_shift`
`parm`	which parameter is to be given a confidence interval. At present there is only one option: the mean of the null distribution. Defaults to "out", referring to the null distribution in objects of class `cyclic_shift`.
`level`	the confidence level required.
`...`	further arguments to `quantile`

Value

A dataframe with the following columns:

- lowerCI: A numeric column with the lowest confidence interval value.
- upperCI: A numeric column with the highest confidence interval value.
- nullMean: A numeric column with the average value of the specified test statistic when calculated on a null community.

References

Hallett, Lauren M., Joanna S. Hsu, Elsa E. Cleland, Scott L. Collins, Timothy L. Dickson, Emily C. Farrer, Laureano A. Gherardi, et al. "Biotic Mechanisms of Community Stability Shift along a Precipitation Gradient." Ecology 95, no. 6 (2014): 1693-1700.

Harms, Kyle E., Richard Condit, Stephen P. Hubbell, and Robin B. Foster. "Habitat Associations of Trees and Shrubs in a 50-Ha Neotropical Forest Plot." Journal of Ecology 89, no. 6 (2001): 947-59.

Examples

```
# Calculate a covariance matrix on a null community
data(knz_001d)
a1_cyclic <- cyclic_shift(subset(knz_001d, subplot == "A_1"),
                  time.var = "year",
                  species.var = "species",
                  abundance.var = "abundance",
                  FUN = cov,
```

```
                    bootnumber = 10)

# Return CI on a1_cyclic
confint(a1_cyclic)
```

`cyclic_shift` *Cyclic Shift Permutations*

Description

Performs a user-specified function on a null ecological community created via cyclic shift permutations (Harms et al. 2001, Hallett et al. 2014). The null community is formed by randomly selected different starting years for the time series of each species. This generates a null community matrix in which species abundances vary independently but within-species autocorrelation is maintained. The user-specified function must require a species x time input.

Usage

```
cyclic_shift(df, time.var, species.var, abundance.var, replicate.var = NA,
  FUN, bootnumber)
```

Arguments

`df`	A data frame containing time, species and abundance columns and an optional column of replicates
`time.var`	The name of the time column
`species.var`	The name of the species column
`abundance.var`	The name of the abundance column
`replicate.var`	The name of the replicate column. Defaults to `NA`, indicating no replicates (i.e., data are from a single plot).
`FUN`	A function to calculate on the null community
`bootnumber`	Number of null simulations to run

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables.

Value

The cyclic_shift function returns an S3 object of class "cyclic_shift" and param "out". The length of the "out" param is the number of null iterations as specified by bootnumber. If multiple replicates are specified, null values are averaged among replicates for each interation, but a different cyclic shift permutation is applied to each replicate within an interation.

References

Hallett, Lauren M., Joanna S. Hsu, Elsa E. Cleland, Scott L. Collins, Timothy L. Dickson, Emily C. Farrer, Laureano A. Gherardi, et al. "Biotic Mechanisms of Community Stability Shift along a Precipitation Gradient." Ecology 95, no. 6 (2014): 1693-1700.

Harms, Kyle E., Richard Condit, Stephen P. Hubbell, and Robin B. Foster. "Habitat Associations of Trees and Shrubs in a 50-Ha Neotropical Forest Plot." Journal of Ecology 89, no. 6 (2001): 947-59.

Examples

```
# Calculate a covariance matrix on a null community
data(knz_001d)
a1_cyclic <- cyclic_shift(subset(knz_001d, subplot == "A_1"),
                  time.var = "year",
                  species.var = "species",
                  abundance.var = "abundance",
                  FUN = cov,
                  bootnumber = 10)
```

`cyclic_shift_onerep` *A function to calculate a non-S3 cyclic shift on one replicate*

Description

A function to calculate a non-S3 cyclic shift on one replicate

Usage

```
cyclic_shift_onerep(df, time.var, species.var, abundance.var, FUN, bootnumber)
```

Arguments

`df`	A data frame containing time, species and abundance columns and an optional column of replicates
`time.var`	The name of the time column
`species.var`	The name of the species column
`abundance.var`	The name of the abundance column
`FUN`	A function to calculate on the null community
`bootnumber`	The number of null model iterations returned

Value

out A vector of test statistics calculated on the null community

df_intersect *Create intersected data frames*

Description

Create intersections.

Usage

```
df_intersect(df1, df2, dataname = "species")
```

Arguments

df1	A dataframe
df2	A dataframe
dataname	The name of the column on which the two datasets will be joined and intersected

knz_001d *Data from Konza Prairie, watershed 001d*

Description

Plant composition within multiple replicates at an annually burned tallgrass prairie site in the Konza Prairie LTER, Manhattan KS (Watershed 001d).

Usage

```
data(knz_001d)
```

Format

A data frame with 8768 rows and 4 variables

Details

A data frame containing a column for species, year, subplot and abundance:

- species: A factor column of species sampled
- year: An integer column of sampling time points
- subplot: A factor column of spatial replicates with 20 levels
- abundance: A numeric column of abundance values

Source

Konza Prairie LTER Dataset ID: PVC02, watershed 1D

Collins, S. L. (2000) Disturbance frequency and community stability in native tallgrass prairie. American Naturalist 155:311-325.

lagged_distances *Get lagged distances for a single replicate*

Description

Returns a data frame with two columns, interval and distance. The interval is the number of time steps between two communities, while distance is the euclidean distance of community change within one replicate lagged across invtervals.

Usage

```
lagged_distances(df, time.var, species.var, abundance.var)
```

Arguments

df	data frame to compute the slope of community change for
time.var	The name of the time column from df
species.var	The name of the species column from df
abundance.var	The name of the abundance column from df

Value

a data frame containing of time lags by species distances

lagged_slope *Get slope Returns the slope of community change within one replicate.*

Description

Get slope Returns the slope of community change within one replicate.

Usage

```
lagged_slope(df, time.var, species.var, abundance.var)
```

Arguments

df	data frame to compute the slope of community change for
time.var	The name of the time column from df
species.var	The name of the species column from df
abundance.var	The name of the abundance column from df

Value

a slope of time lags by species distances

`lag_i` *Get lagged values from a distance matrix Get lagged values from distance matrix at value i*

Description

Get lagged values from a distance matrix Get lagged values from distance matrix at value i

Usage

```
lag_i(i, DM, rownums, colnums)
```

Arguments

`i`	the index of the matrix to lag
`DM`	the distance matrix from which lagged values are drawn
`rownums`	number of rows in the distance matrix
`colnums`	number of columns in the distance matrix

Value

the lagged values

`mean_rank_shift` *Mean Rank Shifts*

Description

A measure of the relative change in species rank abundances, which indicates shifts in relative abundances over time (Collins et al. 2008). Mean rank shifts are calculated as the average difference in species' ranks between consecutive time periods, among species that are present across the entire time series.

Usage

```
mean_rank_shift(df, time.var, species.var, abundance.var,
  replicate.var = as.character(NA))
```

Arguments

`df`	A data frame containing time, species and abundance columns and an optional column of replicates
`time.var`	The name of the time column
`species.var`	The name of the species column
`abundance.var`	The name of the abundance column
`replicate.var`	The name of the optional replicate column

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

Value

mean_rank_shift returns a data frame with the following columns:

- time.var_pair: A factor column that returns the two time periods being compared, separated by a dash. The name of this column is the same as the time.var column in the input dataframe followed by "_pair".
- MRS: A numeric column with the mean rank shift values.
- replicate.var: A column that has same name and type as the replicate.var column, if replication is specified.

---

rank_onerep *Function for calculating mean rank shifts*

Description

This is a function that calculates mean rank shifts

Usage

```
rank_onerep(df, time.var, species.var, abundance.var)
```

Arguments

df	dataframe of Community dataset. Must be in 'long' format.
time.var	The time variable
species.var	The species variable
abundance.var	The abundance variable

Value

a dataframe, showing years compared

`rank_shift` *Mean Rank Shifts*

Description

A measure of the relative change in species rank abundances, which indicates shifts in relative abundances over time (Collins et al. 2008). Mean rank shifts are calculated as the average difference in species' ranks between consecutive time periods, among species that are present across the entire time series.

Usage

```
rank_shift(df, time.var, species.var, abundance.var,
  replicate.var = as.character(NA))
```

Arguments

`df`	A data frame containing time, species and abundance columns and an optional column of replicates
`time.var`	The name of the time column
`species.var`	The name of the species column
`abundance.var`	The name of the abundance column
`replicate.var`	The name of the optional replicate column

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

Value

rank_shift returns a data frame with the following columns:

- time.var_pair: A factor column that returns the two time periods being compared, separated by a dash. The name of this column is the same as the time.var column in the input dataframe followed by "_pair".
- MRS: A numeric column with the mean rank shift values.
- replicate.var: A column that has same name and type as the replicate.var column, if replication is specified.

References

Collins, Scott L., Katharine N. Suding, Elsa E. Cleland, Michael Batty, Steven C. Pennings, Katherine L. Gross, James B. Grace, Laura Gough, Joe E. Fargione, and Christopher M. Clark. (2008) "Rank clocks and plant community dynamics." Ecology 89, no. 12: 3534-41.

Examples

```
# Calculate mean rank shifts within replicates
data(knz_001d)

myoutput <- rank_shift(knz_001d,
                    time.var = "year",
                    species.var = "species",
                    abundance.var = "abundance",
                    replicate.var = "subplot")

# Calculate mean rank shifts for a data frame with no replication

myoutput_singlerep <- rank_shift(subset(knz_001d, subplot=="A_1"),
                          time.var = "year",
                          species.var = "species",
                          abundance.var = "abundance")
```

---

rate_change *Rate of community change over successive time intervals*

Description

Calculates the slope of the differences in species composition within a community over increasing time intervals, which provides a measures of the rate of directional change in community composition. Differences in species composition are characterized by Euclidean distances, which are calculated on pair-wise communities across the entire time series. For example, a data set with six time intervals will have distance values for five one-year time lags (year 1 vs year 2, year 2 vs year 3 ...), four two-year time lags (year 1 vs year 3, year 2 vs year 4 ...) and so forth. These distance values are regressed against the time lag interval. The slope of the regression line is reported as an indication of the rate and direction of compositional change in the community.

Usage

```
rate_change(df, time.var, species.var, abundance.var, replicate.var = NA)
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
species.var	The name of the species column
abundance.var	The name of the abundance column
replicate.var	The name of the optional replicate column

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

The `rate_change` function uses linear regression to relate Euclidean distances to time lag intervals. It is recommended that fit of this relationship be verified using `rate_change_interval`, which returns the full set of community distance values and associated time lag intervals.

Value

The `rate_change` function returns a numeric rate change value unless a replication column is specified in the input data frame. If replication is specified, the function returns a data frame with the following attributes:

- rate_change: A numeric column with the synchrony values.
- replicate.var: A column that shares the same name and type as the replicate.var column in the input data frame.

References

Collins, S. L., Micheli, F. and Hartt, L. 2000. A method to determine rates andpatterns of variability in ecological communities. - Oikos 91: 285-293.

Examples

```
data(knz_001d)
rate_change(knz_001d[knz_001d$subplot=="A_1",],
                       time.var = "year",
                       species.var = "species",
                       abundance.var = "abundance") # for one subplot

rate_change(knz_001d,
             time.var = "year",
             species.var = "species",
             abundance.var = "abundance",
             replicate.var = "subplot") # across all subplots
```

`rate_change_interval` *Differences in community composition over successive time lag intervals*

Description

Calculates the differences in species composition within a community over increasing time intervals. Differences in species composition are characterized by Euclidean distances, which are calculated on pair-wise communities across the entire time series. For example, a data set with 6 time intervals will have distance values for five one-year time lags (year 1 vs year 2, year 2 vs year 3 ...), 4 two-year time lags (year 1 vs year 3, year 2 vs year 4 ...) and so forth. Returns the full set of community distance values and associated time lag intervals.

Usage

```
rate_change_interval(df, time.var, species.var, abundance.var,
  replicate.var = NA)
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
species.var	The name of the species column
abundance.var	The name of the abundance column
replicate.var	The name of the optional replicate column

Value

The rate_change_interval function returns a data frame with the following attributes:

- interval: A numeric column containing the interval length between time periods.
- distance: A numeric column containing the Euclidean distances.
- replicate.var: A column that shares the same name and type as the replicate.var column in the input data frame.

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

References

Collins, S. L., Micheli, F. and Hartt, L. 2000. A method to determine rates andpatterns of variability in ecological communities. - Oikos 91: 285-293.

Examples

```
data(knz_001d)
rate_change_interval(knz_001d[knz_001d$subplot=="A_1",],
                     time.var = "year",
                     species.var = "species",
```

```
                      abundance.var = "abundance") # for one subplot


rate_change_interval(knz_001d,
                      time.var = "year",
                      species.var = "species",
                      abundance.var = "abundance",
                       replicate.var = "subplot") # across all subplots
```

`shuffle_community` *A function to generate a community dataframe with a random start time for each species*

Description

A function to generate a community dataframe with a random start time for each species

Usage

```
shuffle_community(comdat)
```

Arguments

`comdat`	A community dataframe

Value

rand.comdat A randomized community dataframe

`stability_onerep` *A function to calculate species synchrony over time within one replicate*

Description

A function to calculate species synchrony over time within one replicate

Usage

```
stability_onerep(df, x)
```

Arguments

`df`	A dataframe containing x column
`x`	The column to calculate stability on

Value

Stability of x, calculated as the mean/sd

synchrony *Species synchrony*

Description

Calculates the degree synchrony in species abundances within a community over time. Includes the option for two different synchrony metrics. The first, developed by Loreau and de Mazancourt (2008), compares the variance of the aggregated community with the variance of individual components. The second, developed by Gross et al. (2014), compares the average correlation of each individual species with the rest of the aggregated community.

Usage

```
synchrony(df, time.var, species.var, abundance.var, metric = "Loreau",
  replicate.var = NA)
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
species.var	The name of the species column
abundance.var	The name of the abundance column
metric	The synchrony metric to return: • "Loreau": The default metric, calculates synchrony following Loreau and de Mazancourt (2008). • "Gross": Calculates synchrony following Gross et al. (2014).
replicate.var	The name of the optional replicate column

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

Value

The synchrony function returns a numeric synchrony value unless a replication column is specified in the input data frame. If replication is specified, the function returns a data frame with the following attributes:

- synchrony: A numeric column with the synchrony values.
- replicate.var: A column that shares the same name and type as the replicate.var column in the input data frame.

References

Gross, Kevin, Bradley J. Cardinale, Jeremy W. Fox, Andrew Gonzalez, Michel Loreau, H. Wayne Polley, Peter B. Reich, and Jasper van Ruijven. (2014) "Species richness and the temporal stability of biomass production: A new analysis of recent biodiversity experiments." The American Naturalist 183, no. 1: 1-12. doi:10.1086/673915.

Loreau, Michel, and Claire de Mazancourt. (2008) "Species synchrony and its drivers: Neutral and nonneutral community dynamics in fluctuating environments." The American Naturalist 172, no. 2: E48-66. doi:10.1086/589746.

Examples

```
data(knz_001d)
synchrony(knz_001d[knz_001d$subplot=="A_1",],
                    time.var = "year",
                    species.var = "species",
                    abundance.var = "abundance") # for one subplot
## Not run:
synchrony(knz_001d,
             time.var = "year",
             species.var = "species",
             abundance.var = "abundance",
             replicate.var = "subplot") # across all subplots

synchrony(knz_001d,
             time.var = "year",
             species.var = "species",
             abundance.var = "abundance",
             replicate.var = "subplot",
             metric="Gross") # With Gross et al. (2014) metric.

## End(Not run)
```

`synch_onerep` *A function to calculate species synchrony over time within one replicate*

Description

A function to calculate species synchrony over time within one replicate

Usage

```
synch_onerep(df, time.var, species.var, abundance.var, metric = "Loreau")
```

Arguments

df	A dataframe containing rep, time, species and abundance columns
time.var	The name of the time column from df
species.var	The name of the species column from df
abundance.var	The name of the abundance column from df
metric	The synchrony metric to return. The default, "Loreau", returns synchrony as calculated by Loreau and de Mazancourt 2008. The alternative, "Gross", returns synchrony as calculated by Gross et al. 2014

Value

output The degree of species synchrony. If "Loreau", 1 is perfect synchrony and 0 is perfect asynchrony. If "Gross", 1 is perfect synchrony and -1 is perfect asynchrony.

---

transpose_community *Convert from a longform abundance dataframe to a time by species dataframe.*

Description

Convert from a longform abundance dataframe to a time by species dataframe.

Usage

```
transpose_community(df, time.var, species.var, abundance.var)
```

Arguments

df	A dataframe containing time.var, species.var and abundance.var columns
time.var	The name of the time column from df
species.var	The name of the species column from df
abundance.var	The name of the abundance column from df

Value

A dataframe of species abundances x time

turnover *Species Turnover*

Description

Computes species turnover between time periods as the proportion of species either gained or lost relative to the total number of species observed across both time periods. Includes an option to compute turnover as just the proportion of species gained (i.e., "appearances") or lost (i.e., "disappearances").

Usage

```
turnover(df, time.var, species.var, abundance.var, replicate.var = NA,
  metric = "total")
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
species.var	The name of the species column
abundance.var	The name of the abundance column
replicate.var	The name of the optional replicate column
metric	The turnover metric to return: • total: The default metric, calculates summed appearances and disappearances relative to total species richness across both time periods. • appearance: Calculates the number of species that appeared in the second time period relative to total species richness across both time periods. • disappearance: Calculates the number of species that disappeared in the second time period relative to total species richness across both time periods.

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

Value

The turnover function returns a data frame with the following attributes:

- turnover: A numeric column with the turnover values. The name of this column is the same as the specified metric (default is "total").

- time.var: A column containing the second time point; the name and type of this column is the same as the time.var column in the input dataframe.
- replicate.var: A column that has same name and type as the replicate.var column, if replication is specified.

References

Cleland, Elsa E., Scott L. Collins, Timothy L. Dickson, Emily C. Farrer, Katherine L. Gross, Laureano A. Gherardi, Lauren M. Hallett, et al. (2013) "Sensitivity of grassland plant community composition to spatial vs. temporal variation in precipitation." Ecology 94, no. 8: 1687-96.

Examples

```
data(knz_001d)

# Calculate relative total turnover within replicates
total.res <- turnover(df=knz_001d,
              time.var = "year",
              species.var = "species",
              abundance.var = "abundance",
              replicate.var="subplot")

# Calculate relative species appearances within replicates
appear.res <- turnover(df=knz_001d,
              time.var = "year",
              species.var = "species",
              abundance.var = "abundance",
              replicate.var="subplot",
              metric="appearance")

# Calculate relative species disappearances within replicates
disappear.res <- turnover(df=knz_001d,
              time.var = "year",
              species.var = "species",
              abundance.var = "abundance",
              replicate.var="subplot",
              metric="disappearance")
```

---

turnover_allyears *A function to calculate species turnover between years*

Description

A function to calculate species turnover between years

Usage

```
turnover_allyears(df, time.var, species.var, abundance.var,
  metric = c("total", "disappearance", "appearance"))
```

Arguments

df	A dataframe containing time, species and abundance columns
time.var	The name of the time column from df
species.var	The name of the species column from df
abundance.var	The name of the abundance column from df
metric	The turnover metric to return; the default, total, returns summed appearances and disappearances relative to total species richness across both years • appearance: returns the number of appearances in the second year relative to total species richness across both years • disappearance: returns the number of disappearances in the second year relative to the total species richness across both years

Value

output A dataframe containing the specificed turnover metric and year

---

turnover_twoyears *A function to calculate species turnover between two years*

Description

A function to calculate species turnover between two years

Usage

```
turnover_twoyears(d1, d2, species.var, metric = c("total", "disappearance",
  "appearance"))
```

Arguments

d1	A dataframe containing a species column from one year
d2	A dataframe containing a species column from the following year
species.var	The name of the species column in d1 and d2
metric	The turnover metric to return; the default, total, returns summed appearances and disappearances relative to total species richness across both years • appearance: returns the number of appearances in the second year relative to total species richness across both years • disappearance: returns the number of disappearances in the second year relative to the total species richness across both years

Value

output The specificed turnover metric

variance_ratio *Variance Ratio*

Description

Computes the ratio of the variance of aggregate species abundances in a community to the sum of the variances of individual, component species. A variance ratio = 1 indicates that species do not covary, a variance ratio > 1 indicates predominately positive covariance among species and a variance ratio < 1 indicates predominately negative covariance (Schluter 1984).

Includes a cyclic shift null modeling option to test if the variance ratio significantly differs from 1. The null community is created by randomly selecting different starting points for each species' time series, which generates a community in which species abundances vary independently but within-species autocorrelation is maintained (Hallett et al. 2014). This randomization is repeated a user-specific number of times and confidence intervals are reported for the resultant null distribution of variance ratios. If the dataframe includes multiple replicates, the variance ratios for the actual and null communities are averaged within each iteration unless specified otherwise.

Usage

```
variance_ratio(df, time.var, species.var, abundance.var, bootnumber,
  replicate.var = NA, average.replicates = TRUE, level = 0.95, li, ui)
```

Arguments

df	A data frame containing time, species and abundance columns and an optional column of replicates
time.var	The name of the time column
species.var	The name of the species column
abundance.var	The name of the abundance column
bootnumber	The number of null model iterations used to calculated confidence intervals
replicate.var	The name of the (optional) replicate column
average.replicates	If true returns the variance ratio and CIs averaged
level	The confidence level for the null mean
li	(deprecated) lower confidence interval
ui	(deprecated) upper confidence interval across replicates; if false returns the variance ratio and CI for each replicate. Defaults to true.

Details

The input data frame needs to contain columns for time, species and abundance; time.var, species.var and abundance.var are used to indicate which columns contain those variables. If multiple replicates are included in the data frame, that column should be specified with replicate.var. Each replicate

should reflect a single experimental unit - there must be a single abundance value per species within each time point and replicate.

Null model confidence intervals default to the standard lowest 2.5% and upper 97.5% of the null distribution, typically these do not need to be change, but they can be user-modified to set more stringent CIs. @references Hallett, Lauren M., Joanna S. Hsu, Elsa E. Cleland, Scott L. Collins, Timothy L. Dickson, Emily C. Farrer, Laureano A. Gherardi, et al. (2014) "Biotic Mechanisms of Community Stability Shift along a Precipitation Gradient." Ecology 95, no. 6: 1693-1700. doi: 10.1890/13-0895.1

Schluter, Dolph. (1984) "A Variance Test for Detecting Species Associations, with Some Example Applications." Ecology 65, no. 3: 998-1005. doi:10.2307/1938071.

Value

The variance_ratio function returns a data frame with the following attributes:

- VR: A numeric column with the actual variance ratio value.
- lowerCI: A numeric column with the lowest confidence interval value.
- upperCI: A numeric column with the highest confidence interval value.
- nullmean: A numeric column with the average null variance ratio value.
- replicate.var: A column that has same name and type as the replicate.var column, if replication is specified.

Examples

```
data(knz_001d)

# Calculate the variance ratio and CIs averaged within replicates
# Here the null model is repeated once, for final use it is recommended to set a
# large bootnumber (eg, 10000)

res_averagedreplicates <- variance_ratio(knz_001d,
              time.var = "year",
              species.var = "species",
              abundance.var = "abundance",
              bootnumber = 1,
              replicate = "subplot")

#Calculate the variance ratio and CIs for each replicate

res_withinreplicates <- variance_ratio(knz_001d,
              time.var = "year",
              species.var = "species",
              abundance.var = "abundance",
              bootnumber = 1,
              replicate = "subplot",
              average.replicates = FALSE)
```

variance_ratio_longformdata

A function to calculate the variance ratio from a longform dataframe

Description

A function to calculate the variance ratio from a longform dataframe

Usage

```
variance_ratio_longformdata(df, time.var, species.var, abundance.var)
```

Arguments

df	A dataframe containing time.var, replicate.var, species.var and abundance.var columns
time.var	The name of the time.var column from df
species.var	The name of the species.var column from df
abundance.var	The name of the abundance.var column from df

Value

var.ratio The variance ratio of the community

variance_ratio_matrixdata

A function to calculate the variance ratio

Description

A function to calculate the variance ratio

Usage

```
variance_ratio_matrixdata(comdat)
```

Arguments

comdat	A community dataframe

Value

var.ratio The variance ratio of the community

Index