Package ‘taxdiv’

April 1, 2026

Title Taxonomic Diversity Indices Using Deng Entropy

Version 0.1.0

Description Calculates taxonomic diversity indices for ecological community data using Deng entropy framework and classical approaches (Shannon, Simpson, Clarke & Warwick). Provides functions for computing taxonomic distinctness, average taxonomic distinctness (AvTD/Delta+), variation in taxonomic distinctness (VarTD/Lambda+), and Deng entropy-based measures that incorporate taxonomic hierarchy information. Includes tools for constructing taxonomic trees and computing pairwise taxonomic distances.

Imports stats, rlang

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Suggests testthat (>= 3.0.0), knitr, rmarkdown, ggplot2

Config/testthat/edition 3

VignetteBuilder knitr

URL `https://github.com/mgorgoz/taxonomic-diversity-r`, `https://mgorgoz.github.io/taxonomic-diversity-r/`

BugReports `https://github.com/mgorgoz/taxonomic-diversity-r/issues`

NeedsCompilation no

Author Muhammet Murat Gorgoz [aut, cre] (ORCID: <`https://orcid.org/0000-0002-6398-0005`>), Kursad Ozkan [aut] (ORCID: <`https://orcid.org/0000-0002-8526-7243`>), Mehmet Guvenc Negiz [aut] (ORCID: <`https://orcid.org/0009-0008-1507-8712`>)

Maintainer Muhammet Murat Gorgoz <`muratgorgoz350@gmail.com`>

Repository CRAN

Date/Publication 2026-04-01 08:50:02 UTC

Contents

`anatolian_trees` *Anatolian Forest Trees: Multi-Site Species Data*

Description

A data frame containing 20 tree species from Anatolian forests, distributed across three sample plots with varying community compositions. Species abundances follow the Westhoff & van der Maarel (1973) scale (1–9). Taxonomic classification includes seven ranks from species to kingdom.

Usage

```
anatolian_trees
```

Format

A data frame with 33 rows and 9 columns:

Site Sample plot name (character)

Species Binomial species name with underscore separator (character)

Genus Genus (character)

Family Family (character)

Order Order (character)

Class Class (character)

Phylum Phylum / Division (character)

Kingdom Kingdom (character)

Abundance Westhoff abundance value, integer 1–9 (numeric)

Details

The three sites represent different forest types:

Karisik_Orman Mixed forest – both conifers and broadleaves (12 species)

Yaprakli_Orman Broadleaf-dominated forest (13 species)

Konifer_Orman Conifer-dominated forest (8 species)

This dataset can be used directly with `batch_analysis` for multi-site analysis:

```
batch_analysis(anatolian_trees)
```

To extract a single community for use with `ozkan_pto` or `compare_indices`:

```
site1 <- anatolian_trees[anatolian_trees$Site == "Karisik_Orman", ]
community <- setNames(site1$Abundance, site1$Species)
tax_tree  <- site1[, c("Species", "Genus", "Family", "Order",
                        "Class", "Phylum", "Kingdom")]
ozkan_pto(community, tax_tree)
```

References

Westhoff, V. & van der Maarel, E. (1973). The Braun-Blanquet approach. In: R.H. Whittaker (ed.), Ordination and classification of communities. Handbook of Vegetation Science 5, 617–726.

See Also

`batch_analysis` for multi-site analysis, `gazi_comm` and `gazi_gytk` for a single-community example.

Examples

```
data(anatolian_trees)
head(anatolian_trees)

# Multi-site analysis
batch_analysis(anatolian_trees)

# Single site extraction
site1 <- anatolian_trees[anatolian_trees$Site == "Karisik_Orman", ]
comm  <- setNames(site1$Abundance, site1$Species)
tax   <- site1[, c("Species", "Genus", "Family", "Order",
                   "Class", "Phylum", "Kingdom")]
ozkan_pto(comm, tax)
```

`avtd` *Average Taxonomic Distinctness (Delta+)*

Description

Calculates the average taxonomic distinctness (AvTD, Delta+) based on Clarke & Warwick (1998). This is a presence/absence-based measure of the average taxonomic distance between all pairs of species.

Usage

```
avtd(species, tax_tree, weights = NULL)
```

Arguments

`species`	Character vector of species names present in the community (presence-only data).
`tax_tree`	A data frame representing the taxonomic hierarchy.
`weights`	Optional numeric vector of weights for taxonomic levels.

Value

A numeric value representing the average taxonomic distinctness (Delta+).

References

Clarke, K.R. & Warwick, R.M. (1998). A taxonomic distinctness index and its statistical properties. Journal of Applied Ecology, 35, 523-531.

Examples

```
tax <- data.frame(
  Species = c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis",
              "Abies_nordmanniana"),
  Genus = c("Quercus", "Pinus", "Fagus", "Abies"),
  Family = c("Fagaceae", "Pinaceae", "Fagaceae", "Pinaceae"),
  Order = c("Fagales", "Pinales", "Fagales", "Pinales"),
  stringsAsFactors = FALSE
)

spp <- c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis")
avtd(spp, tax)
```

`batch_analysis` *Batch Analysis from a Single Data Frame*

Description

Computes all diversity indices for one or more sample sites from a single data frame (e.g., imported from Excel). The function automatically detects the site column, taxonomic columns, and abundance column, splits the data by site, and returns a summary data frame with species count and 14 diversity indices per site.

Usage

```
batch_analysis(
  data,
  site_column = NULL,
  tax_columns = NULL,
  abundance_column = "Abundance",
  correction = c("none", "miller_madow", "grassberger", "chao_shen"),
  parallel = FALSE,
  n_cores = NULL
)
```

Arguments

`data`	A data frame containing species data. Must include at minimum a species column, at least one taxonomic rank column, and an abundance column. Optionally includes a site/plot column for multi-site analysis.
`site_column`	Character string specifying the name of the site column. If `NULL` (default), the function searches for columns named `"Site"`, `"site"`, `"Alan"`, `"alan"`, `"Plot"`, or `"plot"`. If no such column is found, all data is treated as a single site.

`tax_columns`	Character vector specifying the names of the taxonomic columns (from Species to highest rank). If `NULL` (default), the function auto-detects columns named `"Species"`, `"Genus"`, `"Family"`, `"Order"`, `"Class"`, `"Phylum"`, and `"Kingdom"` (case-insensitive).
`abundance_column`	Character string specifying the name of the abundance column. Default is `"Abundance"` (case-insensitive match).
`correction`	Bias correction for the Shannon index. One of `"none"` (default), `"miller_madow"`, `"grassberger"`, or `"chao_shen"`. Passed to `shannon()`. See `shannon()` for details.
`parallel`	Logical. If `TRUE`, use parallel processing to compute indices for multiple sites concurrently. Default `FALSE`.
`n_cores`	Number of CPU cores to use when `parallel = TRUE`. Default `NULL` uses up to 2 cores (CRAN policy limit).

Details

When no site column is present (or all values are identical), the entire data set is treated as a single community.

The function calculates the following indices per site:

- **Shannon**: Shannon-Wiener entropy (`shannon`)
- **Simpson**: Gini-Simpson index (`simpson`)
- **Delta**: Clarke & Warwick taxonomic diversity (`delta`)
- **Delta_star**: Clarke & Warwick taxonomic distinctness (`delta_star`)
- **AvTD**: Average taxonomic distinctness (`avtd`)
- **VarTD**: Variation in taxonomic distinctness (`vartd`)
- **uTO**: Unweighted taxonomic diversity (Ozkan pTO, all levels)
- **TO**: Weighted taxonomic diversity (Ozkan pTO, all levels)
- **uTO_plus**: Unweighted taxonomic distance (Ozkan pTO, all levels)
- **TO_plus**: Weighted taxonomic distance (Ozkan pTO, all levels)
- **uTO_max**: Unweighted taxonomic diversity (informative levels only)
- **TO_max**: Weighted taxonomic diversity (informative levels only)
- **uTO_plus_max**: Unweighted taxonomic distance (informative levels only)
- **TO_plus_max**: Weighted taxonomic distance (informative levels only)

Value

A data frame with one row per site and columns: `Site`, `N_Species`, `Shannon`, `Simpson`, `Delta`, `Delta_star`, `AvTD`, `VarTD`, `uTO`, `TO`, `uTO_plus`, `TO_plus`, `uTO_max`, `TO_max`, `uTO_plus_max`, `TO_plus_max`.

See Also

`compare_indices` for analysis with pre-built community vectors, `build_tax_tree` for building taxonomic trees manually.

Examples

```
# Single-site data (no Site column)
df <- data.frame(
  Species   = c("sp1", "sp2", "sp3", "sp4"),
  Genus     = c("G1", "G1", "G2", "G2"),
  Family    = c("F1", "F1", "F1", "F2"),
  Order     = c("O1", "O1", "O1", "O1"),
  Abundance = c(10, 20, 15, 5),
  stringsAsFactors = FALSE
)
batch_analysis(df)

# Multi-site data (with Site column)
df2 <- data.frame(
  Site      = c("A", "A", "A", "B", "B", "B"),
  Species   = c("sp1", "sp2", "sp3", "sp1", "sp3", "sp4"),
  Genus     = c("G1", "G1", "G2", "G1", "G2", "G2"),
  Family    = c("F1", "F1", "F1", "F1", "F1", "F2"),
  Order     = c("O1", "O1", "O1", "O1", "O1", "O1"),
  Abundance = c(10, 20, 15, 5, 25, 10),
  stringsAsFactors = FALSE
)
batch_analysis(df2)
```

`build_tax_tree` *Build a Taxonomic Tree from Species Data*

Description

Creates a taxonomic hierarchy data frame from species classification information. This is a convenience function for constructing the `tax_tree` input required by other functions in the package.

Usage

```
build_tax_tree(species, ...)
```

Arguments

`species`	Character vector of species names.
`...`	Named character vectors for each taxonomic rank, in order from lowest to highest (e.g., Genus, Family, Order).

Value

A data frame with species as the first column and taxonomic ranks as subsequent columns.

Examples

```
tree <- build_tax_tree(
  species = c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis"),
  Genus   = c("Quercus", "Pinus", "Fagus"),
  Family  = c("Fagaceae", "Pinaceae", "Fagaceae"),
  Order   = c("Fagales", "Pinales", "Fagales")
)
```

`compare_indices` *Compare All Diversity Indices Side by Side*

Description

Computes all available diversity indices for one or more communities and returns them in a single data frame. Optionally produces a grouped bar plot for visual comparison.

Usage

```
compare_indices(
  communities,
  tax_tree,
  correction = c("none", "miller_madow", "grassberger", "chao_shen"),
  plot = FALSE
)
```

Arguments

`communities`	A named list of community vectors (named numeric), or a single named numeric vector. When a single vector is provided, it is wrapped in a list with name "Community".
`tax_tree`	A data frame representing the taxonomic hierarchy, as produced by `build_tax_tree`.
`correction`	Bias correction for the Shannon index. One of `"none"` (default), `"miller_madow"`, `"grassberger"`, or `"chao_shen"`. Passed to `shannon()`. See `shannon()` for details.
`plot`	Logical. If `TRUE` and **ggplot2** is available, returns a list with both the data frame and a ggplot object. Default is `FALSE`.

Details

The function calculates the following indices:

- **Shannon**: Shannon-Wiener entropy (`shannon`)
- **Simpson**: Gini-Simpson index (`simpson`)
- **Delta**: Clarke & Warwick taxonomic diversity (`delta`)
- **Delta_star**: Clarke & Warwick taxonomic distinctness (`delta_star`)

- **AvTD**: Average taxonomic distinctness (`avtd`)
- **VarTD**: Variation in taxonomic distinctness (`vartd`)
- **uTO**: Unweighted taxonomic diversity (Ozkan pTO, all levels)
- **TO**: Weighted taxonomic diversity (Ozkan pTO, all levels)
- **uTO_plus**: Unweighted taxonomic distance (Ozkan pTO, all levels)
- **TO_plus**: Weighted taxonomic distance (Ozkan pTO, all levels)
- **uTO_max**: Unweighted taxonomic diversity (informative levels)
- **TO_max**: Weighted taxonomic diversity (informative levels)
- **uTO_plus_max**: Unweighted taxonomic distance (informative levels)
- **TO_plus_max**: Weighted taxonomic distance (informative levels)

Value

If `plot = FALSE`, a data frame with communities as rows and indices as columns. If `plot = TRUE`, a list with two elements:

table The data frame of index values.

plot A `ggplot` object showing a grouped bar chart.

See Also

`shannon`, `simpson`, `delta`, `delta_star`, `avtd`, `vartd`, `ozkan_pto`, `pto_components`

Examples

```
tax <- build_tax_tree(
  species = c("sp1", "sp2", "sp3", "sp4"),
  Genus   = c("G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F2"),
  Order   = c("O1", "O1", "O1", "O1")
)

# Single community
comm <- c(sp1 = 10, sp2 = 20, sp3 = 15, sp4 = 5)
compare_indices(comm, tax)

# Multiple communities
comm_list <- list(
  Site_A = c(sp1 = 10, sp2 = 20, sp3 = 15, sp4 = 5),
  Site_B = c(sp1 = 5, sp2 = 5, sp3 = 5, sp4 = 5)
)
compare_indices(comm_list, tax)
```

`delta` *Taxonomic Diversity Index (Delta)*

Description

Calculates the taxonomic diversity index (Delta) from Warwick & Clarke (1995). This is the average weighted path length between every pair of individuals, including same-species pairs (weighted 0).

Usage

```
delta(community, tax_tree, weights = NULL)
```

Arguments

`community`	A named numeric vector of species abundances.
`tax_tree`	A data frame with taxonomic hierarchy.
`weights`	Optional numeric vector of path weights for each taxonomic level. If NULL, a linear scale is used (1, 2, 3, ...).

Details

$$\Delta = \frac{\sum\sum_{i<j} w_{ij} x_i x_j + \sum_i 0 \cdot x_i(x_i - 1)/2}{\sum\sum_{i<j} x_i x_j + \sum_i x_i(x_i - 1)/2}$$

Value

A numeric value representing taxonomic diversity (Delta).

References

Warwick, R.M. & Clarke, K.R. (1995). New 'biodiversity' measures reveal a decrease in taxonomic distinctness with increasing stress. Marine Ecology Progress Series, 129, 301-305.

See Also

`delta_star()` for taxonomic distinctness (excluding same-species), `avtd()` for presence/absence-based AvTD, `ozkan_pto()` for Deng entropy-based alternative.

Examples

```
comm <- c(sp1 = 5, sp2 = 3, sp3 = 3, sp4 = 1, sp5 = 3)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus = c("G1", "G1", "G2", "G2", "G2"),
  Family = c("F1", "F1", "F1", "F2", "F2"),
  stringsAsFactors = FALSE
)
```

```
delta(comm, tax)
```

---

delta_star *Taxonomic Distinctness (Delta*)*

Description

Calculates the taxonomic distinctness (Delta*) from Warwick & Clarke (1995). This is the average weighted path length between individuals of different species only.

Usage

```
delta_star(community, tax_tree, weights = NULL)
```

Arguments

`community`	A named numeric vector of species abundances.
`tax_tree`	A data frame with taxonomic hierarchy.
`weights`	Optional numeric vector of path weights for each taxonomic level. If NULL, a linear scale is used (1, 2, 3, ...).

Details

$$\Delta^* = \frac{\sum\sum_{i<j} w_{ij} x_i x_j}{\sum\sum_{i<j} x_i x_j}$$

Value

A numeric value representing taxonomic distinctness (Delta*).

References

Warwick, R.M. & Clarke, K.R. (1995). New ’biodiversity’ measures reveal a decrease in taxonomic distinctness with increasing stress. Marine Ecology Progress Series, 129, 301-305.

See Also

`delta()` for taxonomic diversity (including same-species), `avtd()` and `vartd()` for presence/absence measures.

Examples

```
comm <- c(sp1 = 5, sp2 = 3, sp3 = 3, sp4 = 1, sp5 = 3)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus = c("G1", "G1", "G2", "G2", "G2"),
  Family = c("F1", "F1", "F1", "F2", "F2"),
  stringsAsFactors = FALSE
)
delta_star(comm, tax)
```

`deng_entropy_level` *Calculate Deng Entropy at a Single Taxonomic Level*

Description

Computes the Deng entropy (Ed) for a given set of group proportions at a specific taxonomic level. This is the core entropy calculation from Deng (2016), which generalizes Shannon entropy through the Dempster-Shafer evidence theory framework.

Usage

```
deng_entropy_level(
  abundances,
  group_sizes = NULL,
  correction = c("none", "miller_madow", "grassberger", "chao_shen")
)
```

Arguments

`abundances`	A numeric vector of abundances for each group (node) at the given taxonomic level.
`group_sizes`	Optional integer vector of focal element sizes (`|Fi|`) for each group. At species level this is NULL (all sizes are 1, reducing to Shannon entropy). At higher taxonomic levels, each value represents the number of species within that group.
`correction`	Bias correction method for Shannon entropy estimation. Only applied at species level (`group_sizes = NULL`). One of `"none"` (default), `"miller_madow"`, `"grassberger"`, or `"chao_shen"`. See `shannon()` for details. A warning is issued if correction is requested with non-NULL `group_sizes`.

Details

The Deng entropy is calculated as:

$$E_d = -\sum_i m(F_i) \ln \frac{m(F_i)}{2^{|F_i|} - 1}$$

At species level, each focal element has cardinality 1, so Deng entropy reduces to Shannon entropy:

$$E_d^S = H = -\sum_i p_i \ln p_i$$

At higher levels (genus, family, etc.), $|F_i|$ equals the number of species within each group, and the mass function is the normalized proportion of total abundance in each group.

Bias correction is only meaningful at the species level where Deng entropy equals Shannon entropy. At higher taxonomic levels the mass function has a different structure and bias-correction formulas do not apply.

Value

A numeric value representing the Deng entropy at that level.

References

Deng, Y. (2016). Deng entropy. Chaos, Solitons & Fractals, 91, 549-553.

See Also

`ozkan_pto()` which uses this function internally, `shannon()` for classical Shannon entropy and bias corrections.

Examples

```
# Shannon entropy (species level, |Fi| = 1 for all)
deng_entropy_level(c(4, 2, 3, 1, 2, 3, 2, 2))

# With bias correction at species level
deng_entropy_level(c(4, 2, 3, 1, 2), correction = "chao_shen")

# Deng entropy at genus level with group sizes
deng_entropy_level(c(9, 3, 7), group_sizes = c(3, 2, 3))
```

`gazi_comm` *Example Community Vector: 8 Anatolian Tree Species*

Description

A named numeric vector of species abundances for a single forest community with 8 Anatolian tree species. Abundance values follow the Westhoff & van der Maarel (1973) scale (1–9). This vector mirrors the hypothetical example in Ozkan (2018).

Usage

```
gazi_comm
```

Format

A named numeric vector with 8 elements. Names are species binomials (underscore-separated); values are integer abundances (1–4).

Details

The species include 3 genera from Pinaceae, 2 from Fagaceae, 1 each from Cupressaceae and Betulaceae, spanning 2 orders (Pinales, Fagales).

Pair with `gazi_gytk` for analysis:

```
ozkan_pto(gazi_comm, gazi_gytk)
compare_indices(gazi_comm, gazi_gytk)
```

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336–346.

See Also

`gazi_gytk` for the matching taxonomy, `anatolian_trees` for a multi-site dataset.

Examples

```
data(gazi_comm)
data(gazi_gytk)

# Ozkan pTO
ozkan_pto(gazi_comm, gazi_gytk)

# All indices at once
compare_indices(gazi_comm, gazi_gytk)
```

`gazi_gytk` *Example Taxonomy: 8 Anatolian Tree Species*

Description

A data frame containing the taxonomic hierarchy for the 8 species in `gazi_comm`, with 3 taxonomic ranks (Genus, Family, Order). This compact taxonomy table is designed for quick demonstrations and unit testing.

Usage

```
gazi_gytk
```

Format

A data frame with 8 rows and 4 columns:

Species Binomial species name (character)

Genus Genus (character)

Family Family (character)

Order Order (character)

Details

The taxonomy represents:

- 8 genera: Pinus, Cedrus, Quercus, Fagus, Juniperus, Carpinus
- 4 families: Pinaceae (3 spp), Fagaceae (3 spp), Cupressaceae (1), Betulaceae (1)
- 2 orders: Pinales (4 spp), Fagales (4 spp)

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336–346.

See Also

`gazi_comm` for the matching community vector, `build_tax_tree` for building custom taxonomies.

Examples

```
data(gazi_gytk)
gazi_gytk

# Compute taxonomic distance matrix
tax_distance_matrix(gazi_gytk)
```

`ozkan_pto` *Calculate Ozkan's Taxonomic Diversity Index (pTO)*

Description

Computes the four components of the Deng entropy-based taxonomic diversity measure proposed by Ozkan (2018): weighted/unweighted taxonomic diversity (TO, uTO) and weighted/unweighted taxonomic distance (TO+, uTO+).

Usage

```
ozkan_pto(community, tax_tree, max_level = NULL)
```

Arguments

`community`	A named numeric vector of species abundances. Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks from lowest to highest (e.g., Species, Genus, Family, Order, Class, Phylum, Kingdom).
`max_level`	Integer or `NULL`. Maximum number of taxonomic levels (above Species) to include in the product formula. When `NULL` (default), all available levels are used. When set to `"auto"`, the function automatically detects the highest informative level (where Deng entropy > 0 at nk=0) and truncates the product there. A positive integer limits to that many levels (e.g., `max_level = 2` uses only Genus and Family).

Details

The method uses the slicing procedure from Ozkan (2018). At each slice (nk = 0, 1, ..., n_s), species with abundance <= nk are removed. The surviving species receive EQUAL weight (1/count) — abundance information enters indirectly through which species survive each slice.

Deng entropy at each taxonomic level is computed using these equal proportions, where the mass function m(Fi) = count_in_group / total_count and |Fi| = number of species in that taxonomic group.

The core product formula at each slice is:

$$\prod_{i=1}^{L}\left(w_i\left(\frac{(e^{E_d^S})^2}{e^{E_d^i}}+1\right)\right)$$

where E_d^S is the Deng entropy at species level and E_d^i is the Deng entropy at level i, computed using presence/absence (equal weight) proportions.

pTO+ (taxonomic distance) uses only the nk=0 slice:

$$pT_O^+ = \ln\prod_{i=1}^{L}\left(w_i\left(\frac{(e^{E_d^S})^2}{e^{E_d^i}}+1\right)\right)$$

pTO (taxonomic diversity) aggregates across all slices:

$$pT_O = \ln\left(\frac{\sum_{k=0}^{n_s}(n_s-n_k)\prod_{i=1}^{L}\left(w_i\left(\frac{(e^{E_d^S})^2}{e^{E_d^i}}+1\right)\right)}{n_s+\sum n_k}\right)$$

Value

A named list with components:

uTO Unweighted taxonomic diversity (all levels)

TO Weighted taxonomic diversity (all levels)

uTO_plus Unweighted taxonomic distance (all levels)

TO_plus	Weighted taxonomic distance (all levels)
uTO_max	Unweighted taxonomic diversity (max informative level)
TO_max	Weighted taxonomic diversity (max informative level)
uTO_plus_max	Unweighted taxonomic distance (max informative level)
TO_plus_max	Weighted taxonomic distance (max informative level)
Ed_levels	Deng entropy at each taxonomic level (nk=0 slice)
max_informative_level	Integer: highest level with Ed > 0

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346. DOI: 10.18182/tjf.441061

Deng, Y. (2016). Deng entropy. Chaos, Solitons & Fractals, 91, 549-553.

See Also

`deng_entropy_level()` for the core Deng entropy calculation, `pto_components()` for a convenience wrapper returning a named vector, `delta()` and `avtd()` for Clarke & Warwick alternatives.

Examples

```
# Simple example with 5 species
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
ozkan_pto(comm, tax)

# With auto max-level detection
ozkan_pto(comm, tax, max_level = "auto")
```

`ozkan_pto_full` *Full Ozkan pTO Pipeline (Islem 1 + 2 + 3)*

Description

Runs the complete Ozkan taxonomic diversity analysis pipeline: jackknife (Islem 1), stochastic resampling (Islem 2), and sensitivity analysis (Islem 3), returning the maximum values across all three runs. This is equivalent to running all three steps in the Excel macro sequentially.

Usage

```
ozkan_pto_full(community, tax_tree, n_iter = 101L, seed = NULL)
```

Arguments

`community`	A named numeric vector of species abundances. Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks.
`n_iter`	Number of stochastic iterations for Run 2 and Run 3 (default: 101, minimum: 101).
`seed`	Optional random seed for reproducibility. If provided, Run 2 uses this seed and Run 3 uses `seed + 1` to ensure independent randomness.

Details

This function implements the full Excel macro pipeline in a single call:

1. **Islem 1**: Leave-one-out jackknife to identify contributing (happy) vs non-contributing (unhappy) species, plus deterministic pTO calculation.
2. **Islem 2**: Stochastic resampling – unhappy species are always included, happy species get 50\
3. **Islem 3**: Sensitivity analysis – unhappy species get $P = (S - 1)/S$, happy species get a data-driven probability.
4. **Final result**: Maximum values across all three runs.

Value

A named list with components:

uTO_plus Final maximum uTO+ across all 3 runs

TO_plus Final maximum TO+ across all 3 runs

uTO Final maximum uTO across all 3 runs

TO Final maximum TO across all 3 runs

run1 Deterministic pTO result (from `ozkan_pto()`)

run2 Full Run 2 result (from `ozkan_pto_resample()`)

run3 Full Run 3 result (from `ozkan_pto_sensitivity()`)

jackknife Jackknife result with species classifications

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346. DOI: 10.18182/tjf.441061

See Also

`ozkan_pto()` for deterministic calculation only, `ozkan_pto_resample()` for Run 2 only, `ozkan_pto_sensitivity()` for Run 3 only, `ozkan_pto_jackknife()` for jackknife only.

Examples

```
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
set.seed(42)
result <- ozkan_pto_full(comm, tax, n_iter = 101)
result$uTO_plus  # Final maximum uTO+
result$TO_plus   # Final maximum TO+
```

`ozkan_pto_jackknife` *Jackknife Analysis for Ozkan's pTO Index (Islem 1 / Run 1)*

Description

Implements the leave-one-out jackknife procedure from the Ozkan Excel macro (Islem 1). Removes each species one at a time, recalculates pTO, and identifies "happy" (contributing) and "unhappy" (non-contributing) species. A species is "happy" if its removal decreases the pTO index, indicating it positively contributes to the community's taxonomic diversity.

Usage

```
ozkan_pto_jackknife(community, tax_tree, component = "uTO_plus")
```

Arguments

`community`	A named numeric vector of species abundances. Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks.
`component`	Character string specifying which pTO component to use for the happy/unhappy classification. One of `"uTO_plus"` (default), `"TO_plus"`, `"uTO"`, or `"TO"`.

Details

The jackknife procedure follows the Excel macro's Islem 1 logic:

1. Compute pTO for the full community.
2. For each species i, remove it and compute pTO for the remaining community (leave-one-out).
3. Compare each leave-one-out result against the full-community value.
4. If removing species i DECREASES the specified component (pTO becomes smaller), species i is classified as "happy" (contributing).

5. If removing species i does NOT decrease the component, species i is classified as "unhappy" (non-contributing).

The happy/unhappy classification is used by `ozkan_pto_resample()` (Islem 2) and `ozkan_pto_sensitivity()` (Islem 3) to apply different resampling probabilities to each species category.

Value

A named list with components:

full_result The `ozkan_pto()` result for the full community

jackknife_results Data frame with leave-one-out results per species

species_status Named logical vector: `TRUE` = happy (contributing), `FALSE` = unhappy (non-contributing)

n_happy Number of happy species

n_unhappy Number of unhappy species

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346. DOI: 10.18182/tjf.441061

See Also

`ozkan_pto()` for the core calculation, `ozkan_pto_resample()` for Run 2, `ozkan_pto_full()` for the full 3-run pipeline.

Examples

```
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
jk <- ozkan_pto_jackknife(comm, tax)
jk$species_status   # Which species are happy (contributing)?
jk$n_happy           # How many happy species?
```

`ozkan_pto_resample` *Stochastic Resampling of Ozkan's pTO Index (Islem 2 / Run 2)*

Description

Implements the stochastic resampling procedure from Ozkan's Excel macro (Islem 2). First performs a jackknife (Islem 1) to identify "happy" (contributing) and "unhappy" (non-contributing) species, then runs stochastic resampling where unhappy species are always included and happy species are randomly included with 50\

Usage

```
ozkan_pto_resample(community, tax_tree, n_iter = 101L, seed = NULL)
```

Arguments

`community`	A named numeric vector of species abundances. Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks.
`n_iter`	Number of stochastic iterations to run (default: 101). Must be >= 101.
`seed`	Optional random seed for reproducibility (default: NULL).

Details

The algorithm follows the Excel macro's Islem 1 + Islem 2 logic:

1. Run jackknife (`ozkan_pto_jackknife()`) to classify each species as happy or unhappy.
2. Iteration 1: Use the original community (deterministic).
3. Iterations 2..n_iter: For each species:
    - Unhappy species (AA = 0): always included with original abundance.
    - Happy species (AA > 0): randomly included (50\ or excluded. Uses `RANDBETWEEN(0,1) * abundance`.
4. Return the maximum of each component across all iterations.

Value

A named list with components:

uTO_plus_max Maximum unweighted taxonomic distance across iterations

TO_plus_max Maximum weighted taxonomic distance across iterations

uTO_max Maximum unweighted taxonomic diversity across iterations

TO_max Maximum weighted taxonomic diversity across iterations

uTO_plus_det Deterministic uTO+ (first iteration, same as `ozkan_pto()`)

TO_plus_det Deterministic TO+ (first iteration)

uTO_det Deterministic uTO (first iteration)

TO_det Deterministic TO (first iteration)

n_iter Number of iterations performed

species_status Named logical vector from jackknife (`TRUE` = happy)

jackknife Full jackknife result from `ozkan_pto_jackknife()`

n_positive Number of iterations with positive uTO+

iteration_results Data frame with all iteration results

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346. DOI: 10.18182/tjf.441061

See Also

`ozkan_pto_jackknife()` for the jackknife step, `ozkan_pto_sensitivity()` for Run 3, `ozkan_pto_full()` for the full pipeline.

Examples

```
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
set.seed(42)
result <- ozkan_pto_resample(comm, tax, n_iter = 101)
result$species_status  # Happy/unhappy classification
```

`ozkan_pto_sensitivity` *Sensitivity Analysis of Ozkan's pTO Index (Islem 3 / Run 3)*

Description

Implements the sensitivity analysis procedure from Ozkan's Excel macro (Islem 3). Uses the jackknife results from Run 2 to apply species-specific inclusion probabilities: unhappy species get $P = (S - 1)/S$, happy species get a data-driven probability derived from Run 2 iteration results.

Usage

```
ozkan_pto_sensitivity(
  community,
  tax_tree,
  run2_result,
  n_iter = NULL,
  seed = NULL
)
```

Arguments

`community`	A named numeric vector of species abundances.
`tax_tree`	A data frame with taxonomic hierarchy.
`run2_result`	The result from `ozkan_pto_resample()`.
`n_iter`	Number of iterations (default: same as Run 2).
`seed`	Optional random seed for reproducibility.

Details

The algorithm follows the Excel macro's Islem 3 logic:

For each species, the inclusion probability depends on its jackknife classification from Islem 1:

- **Unhappy species** (AA = 0, non-contributing): Included with probability $(S-1)/S$, where S is total species count. In the Excel formula: `IF(RANDBETWEEN(1, S) > 1, H2, 0)`.
- **Happy species** (AA > 0, contributing): Included with probability derived from Run 2 results. In the Excel formula: `IF(L25 >= RANDBETWEEN(0, K22), H2, 0)`, where L25 is a summary score from Run 2 and K22 is the iteration count.

The happy species probability is computed as:

$$P_{happy} = \frac{\max(0, N_{positive} - S) + 1}{N_{iter} + 1}$$

where $N_{positive}$ is the number of Run 2 iterations that produced a positive uTO+ value and S is the species count.

The maximum pTO across all three runs (Run 1, 2, 3) is the final result.

Value

A named list with components:

uTO_plus_max Maximum uTO+ across Run 1, 2, and 3

TO_plus_max Maximum TO+ across all runs

uTO_max Maximum uTO across all runs

TO_max Maximum TO across all runs

run3_uTO_plus_max Maximum uTO+ from Run 3 only

run3_TO_plus_max Maximum TO+ from Run 3 only

run3_uTO_max Maximum uTO from Run 3 only

run3_TO_max Maximum TO from Run 3 only

n_iter Number of iterations performed

species_probs Named numeric vector of inclusion probabilities

prob_happy Probability used for happy species

prob_unhappy Probability used for unhappy species

iteration_results Data frame with all Run 3 iteration results

References

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346. DOI: 10.18182/tjf.441061

See Also

`ozkan_pto_resample()` for Run 2, `ozkan_pto_full()` for the full pipeline.

Examples

```
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
set.seed(42)
run2 <- ozkan_pto_resample(comm, tax, n_iter = 101)
ozkan_pto_sensitivity(comm, tax, run2, n_iter = 101)
```

`plot_bubble` *Bubble Chart of Species Contributions to Diversity*

Description

Creates a bubble chart showing each species' abundance (x-axis), average taxonomic distance to other species (y-axis), and relative contribution to the community (bubble size). Species that are both abundant and taxonomically distant from others contribute most to overall taxonomic diversity.

Usage

```
plot_bubble(community, tax_tree, color_by = NULL, title = NULL)
```

Arguments

`community`	Named numeric vector of species abundances.
`tax_tree`	A data frame representing the taxonomic hierarchy, as produced by `build_tax_tree`.
`color_by`	Character string specifying which taxonomic rank to use for coloring bubbles. Must match a column name in `tax_tree`. If `NULL` (default), the highest available rank is used.
`title`	Optional character string for the plot title.

Details

For each species i, the average taxonomic distance is calculated as:

$$\bar{\omega}_i = \frac{1}{S-1} \sum_{j \neq i} \omega_{ij}$$

where ω_{ij} is the pairwise taxonomic distance and S is the number of species. Bubble size represents the product of relative abundance and average distance, indicating each species' contribution to overall taxonomic diversity.

Value

A `ggplot` object.

See Also

`tax_distance_matrix`, `compare_indices`

Examples

```
comm <- c(sp1 = 25, sp2 = 18, sp3 = 30, sp4 = 12, sp5 = 8)
tax <- build_tax_tree(
  species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G2", "G2", "G3"),
  Family  = c("F1", "F1", "F1", "F2", "F2"),
  Order   = c("O1", "O1", "O1", "O1", "O1")
)
plot_bubble(comm, tax)
```

`plot_funnel` *Funnel Plot for AvTD/VarTD*

Description

Produces a Clarke & Warwick style funnel plot showing expected confidence limits for Average Taxonomic Distinctness (AvTD) and/or Variation in Taxonomic Distinctness (VarTD) as a function of species richness. Observed site values can be overlaid to assess whether they fall within or outside the expected range.

Usage

```
plot_funnel(
  sim_result,
  observed = NULL,
  index = c("avtd", "vartd"),
  title = NULL,
  point_labels = TRUE,
  mean_color = "darkblue",
  ci_color = "steelblue"
)
```

Arguments

`sim_result`	A `td_simulation` object returned by `simulate_td()`.
`observed`	Optional data frame with columns `site` (character), `s` (integer, species richness), and `value` (numeric, observed AvTD or VarTD). Points are plotted on the funnel.

`index`	Which index to plot when `sim_result` contains both: `"avtd"` (default) or `"vartd"`.
`title`	Optional plot title. If `NULL`, generated automatically.
`point_labels`	Logical; if `TRUE` (default), label observed points with site names.
`mean_color`	Color of the mean line (default: `"darkblue"`).
`ci_color`	Fill color of the confidence band (default: `"steelblue"`).

Details

The funnel shape arises because small samples (low S) have greater random variation in AvTD/VarTD, producing wider confidence bands. As S approaches the full species pool, the band narrows.

Observed points falling below the lower confidence limit suggest the community has lower taxonomic breadth than expected by chance, potentially indicating environmental stress or biotic homogenisation.

Requires the **ggplot2** package.

Value

A ggplot object.

References

Clarke, K.R. & Warwick, R.M. (1998). A taxonomic distinctness index and its statistical properties. Journal of Applied Ecology, 35, 523-531.

See Also

`simulate_td()` for generating the simulation, `avtd()` and `vartd()` for the underlying calculations.

Examples

```
tax <- data.frame(
  Species = paste0("sp", 1:10),
  Genus   = rep(c("G1", "G2", "G3", "G4", "G5"), each = 2),
  Family  = rep(c("F1", "F1", "F2", "F2", "F3"), each = 2),
  Order   = rep(c("O1", "O1", "O2", "O2", "O2"), each = 2),
  stringsAsFactors = FALSE
)
sim <- simulate_td(tax, n_sim = 99, seed = 42)

# Basic funnel plot
plot_funnel(sim)

# With observed sites
obs <- data.frame(
  site  = c("Site_A", "Site_B"),
  s     = c(5, 8),
  value = c(2.5, 1.8)
)
plot_funnel(sim, observed = obs)
```

`plot_heatmap` *Plot Taxonomic Distance Heatmap*

Description

Visualizes the pairwise taxonomic distance matrix as a colored heatmap using ggplot2. Species are ordered by hierarchical clustering so that taxonomically similar species appear adjacent.

Usage

```
plot_heatmap(
  tax_tree,
  label_size = 3,
  title = NULL,
  low_color = "white",
  high_color = "#B22222"
)
```

Arguments

`tax_tree`	A data frame representing the taxonomic hierarchy, as produced by `build_tax_tree`.
`label_size`	Numeric value controlling the size of species labels. Default is 3.
`title`	Optional character string for the plot title.
`low_color`	Color for the lowest distance (most similar). Default is `"white"`.
`high_color`	Color for the highest distance (most distant). Default is `"#B22222"` (firebrick red).

Details

The heatmap displays the full symmetric distance matrix computed by `tax_distance_matrix`. The diagonal (self-distance = 0) appears in the lowest color. Species are reordered using hierarchical clustering (UPGMA) to reveal taxonomic groupings visually.

Value

A `ggplot` object.

See Also

`tax_distance_matrix`, `plot_taxonomic_tree`

Examples

```
tax <- build_tax_tree(
  species = c("sp1", "sp2", "sp3", "sp4"),
  Genus   = c("G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F2")
)
plot_heatmap(tax)
```

`plot_iteration` *Plot pTO Iteration Results from Run 2 or Run 3*

Description

Visualizes the iteration-by-iteration pTO values from stochastic resampling (Run 2) or sensitivity analysis (Run 3). Each iteration's value is shown as a point, with the deterministic (Run 1) value displayed as a horizontal reference line.

Usage

```
plot_iteration(resample_result, component = "TO", title = NULL)
```

Arguments

`resample_result`	The list returned by `ozkan_pto_resample` (Run 2) or `ozkan_pto_sensitivity` (Run 3).
`component`	Character string specifying which pTO component to plot. One of `"uTO"`, `"TO"`, `"uTO_plus"`, `"TO_plus"`. Default is `"TO"`.
`title`	Optional character string for the plot title.

Details

The plot includes three visual elements:

- **Grey points**: Individual iteration values
- **Red dashed line**: Deterministic (Run 1) value
- **Blue dashed line**: Maximum value across all iterations

This helps assess how stochastic species removal affects the pTO index and whether the maximum exceeds the deterministic value.

Value

A `ggplot` object showing iteration values as points, the deterministic value as a dashed red line, and the maximum value as a dashed blue line.

See Also

ozkan_pto_resample, ozkan_pto_sensitivity

Examples

```
comm <- c(sp1 = 10, sp2 = 20, sp3 = 15, sp4 = 5)
tax <- build_tax_tree(
  species = paste0("sp", 1:4),
  Genus = c("G1", "G1", "G2", "G2"),
  Family = c("F1", "F1", "F1", "F2")
)
res <- ozkan_pto_resample(comm, tax, n_iter = 101, seed = 42)
plot_iteration(res, component = "TO")
```

`plot_radar` *Radar (Spider) Chart for Multi-Community Index Comparison*

Description

Creates a radar chart comparing diversity indices across multiple communities. Each axis represents a different index, and each community is drawn as a colored polygon. Values are normalized to 0-1 scale so that indices with different magnitudes can be compared visually.

Usage

```
plot_radar(communities, tax_tree, indices = NULL, title = NULL)
```

Arguments

`communities`	A named list of community vectors (named numeric).
`tax_tree`	A data frame representing the taxonomic hierarchy, as produced by `build_tax_tree`.
`indices`	Character vector specifying which indices to include. Default is all 10 indices. Available: `"Shannon"`, `"Simpson"`, `"Delta"`, `"Delta_star"`, `"AvTD"`, `"VarTD"`, `"uTO"`, `"TO"`, `"uTO_plus"`, `"TO_plus"`.
`title`	Optional character string for the plot title.

Details

Each index value is normalized using min-max scaling across the communities being compared:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

If all communities have the same value for an index (i.e., $x_{max} = x_{min}$), the normalized value is set to 0.5.

The radar chart is built using polar coordinates in ggplot2. Each community appears as a colored polygon overlay, making it easy to spot which community scores higher on which indices.

Value

A `ggplot` object.

See Also

`compare_indices` for tabular comparison

Examples

```
tax <- build_tax_tree(
  species = c("sp1", "sp2", "sp3", "sp4"),
  Genus   = c("G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F2")
)
comms <- list(
  Site_A = c(sp1 = 10, sp2 = 20, sp3 = 15, sp4 = 5),
  Site_B = c(sp1 = 5, sp2 = 5, sp3 = 5, sp4 = 5)
)
plot_radar(comms, tax)
```

`plot_rarefaction` *Plot Taxonomic Rarefaction Curve*

Description

Visualises a rarefaction curve with confidence intervals using ggplot2. Accepts output from `rarefaction_taxonomic()`.

Usage

```
plot_rarefaction(
  rare_result,
  title = NULL,
  xlab = "Sample Size (individuals)",
  ylab = NULL,
  ci_color = "steelblue",
  line_color = "darkblue"
)
```

Arguments

`rare_result`	A data frame returned by `rarefaction_taxonomic()`.
`title`	Optional plot title. If `NULL`, an automatic title is generated based on the index used.
`xlab`	Label for the x-axis (default: `"Sample Size (individuals)"`).
`ylab`	Label for the y-axis. If `NULL`, determined from the index.
`ci_color`	Fill color for the confidence interval ribbon (default: `"steelblue"`).
`line_color`	Color of the mean line (default: `"darkblue"`).

Details

The plot shows the mean diversity value at each sample size as a solid line, surrounded by a shaded ribbon representing the bootstrap confidence interval. A vertical dashed line marks the total sample size (full data).

Requires the **ggplot2** package.

Value

A ggplot object.

See Also

`rarefaction_taxonomic()` for computing the rarefaction curve.

Examples

```
comm <- c(sp1 = 10, sp2 = 5, sp3 = 8, sp4 = 2, sp5 = 3)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G2", "G2", "G3"),
  Family  = c("F1", "F1", "F1", "F2", "F2"),
  stringsAsFactors = FALSE
)
rare <- rarefaction_taxonomic(comm, tax, index = "shannon", n_boot = 50)
plot_rarefaction(rare)
```

`plot_taxonomic_tree` *Plot Taxonomic Tree as a Dendrogram*

Description

Visualizes a taxonomic hierarchy as a dendrogram (tree diagram) using ggplot2. The function converts the taxonomic distance matrix into a hierarchical clustering object and renders it as a horizontal dendrogram with species labels colored by a chosen taxonomic rank.

Usage

```
plot_taxonomic_tree(
  tax_tree,
  community = NULL,
  color_by = NULL,
  label_size = 3,
  title = NULL
)
```

Arguments

`tax_tree`	A data frame representing the taxonomic hierarchy, as produced by `build_tax_tree`. First column must be species names, subsequent columns are taxonomic ranks from lowest to highest (e.g., Genus, Family, Order).
`community`	Optional named numeric vector of species abundances. Names must match species in `tax_tree`. When provided, abundance values are shown next to species labels.
`color_by`	Character string specifying which taxonomic rank to use for coloring species labels. Must match a column name in `tax_tree`. If `NULL` (default), the highest available rank is used.
`label_size`	Numeric value controlling the size of species labels. Default is 3.
`title`	Optional character string for the plot title. If `NULL` (default), no title is displayed.

Details

The dendrogram is constructed from the pairwise taxonomic distance matrix (computed via `tax_distance_matrix`) using hierarchical clustering (`hclust` with `method = "average"`). Branch heights reflect taxonomic distance: species in the same genus merge at the lowest level, while species in different orders merge at the highest level.

When a `community` vector is provided, species labels are annotated with abundance values in parentheses, e.g., "Quercus_coccifera (25)".

This function requires the **ggplot2** package. If ggplot2 is not installed, the function will stop with an informative error message.

The clustering method used is UPGMA (Unweighted Pair Group Method with Arithmetic Mean), which is standard for taxonomic classification trees where branch lengths represent average distances between groups.

Value

A `ggplot` object that can be further customized with ggplot2 functions (e.g., `+ theme()`, `+ labs()`).

References

Clarke, K.R. & Warwick, R.M. (1998). A taxonomic distinctness index and its statistical properties. *Journal of Applied Ecology*, 35, 523–531.

See Also

`build_tax_tree` for creating the taxonomy input, `tax_distance_matrix` for the underlying distance calculation.

Examples

```
# Build a simple taxonomic tree
tax <- build_tax_tree(
  species = c("Quercus_robur", "Quercus_petraea", "Pinus_nigra",
              "Pinus_brutia", "Juniperus_excelsa"),
```

```
  Genus   = c("Quercus", "Quercus", "Pinus", "Pinus", "Juniperus"),
  Family  = c("Fagaceae", "Fagaceae", "Pinaceae", "Pinaceae", "Cupressaceae"),
  Order   = c("Fagales", "Fagales", "Pinales", "Pinales", "Pinales")
)

# Basic dendrogram
plot_taxonomic_tree(tax)

# Color by Family, with abundance info
comm <- c(Quercus_robur = 25, Quercus_petraea = 18,
          Pinus_nigra = 30, Pinus_brutia = 12,
          Juniperus_excelsa = 8)
plot_taxonomic_tree(tax, community = comm, color_by = "Family")
```

`pto_components` *Calculate All Eight pTO Components (Convenience Wrapper)*

Description

Returns a named numeric vector with all eight Ozkan (2018) components: four using all taxonomic levels and four using only the informative levels (max version), matching the Excel macro's Run 1+2+3 output.

Usage

```
pto_components(community, tax_tree)
```

Arguments

`community`	A named numeric vector of species abundances. Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks from lowest to highest (e.g., Species, Genus, Family, Order, Class, Phylum, Kingdom).

Value

A named numeric vector with eight elements: `uTO`, `TO`, `uTO_plus`, `TO_plus`, `uTO_max`, `TO_max`, `uTO_plus_max`, `TO_plus_max`.

See Also

`ozkan_pto()` for the full result including per-level entropy.

Examples

```
comm <- c(sp1 = 4, sp2 = 2, sp3 = 3, sp4 = 1, sp5 = 2)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G1", "G2", "G2"),
  Family  = c("F1", "F1", "F1", "F1", "F1"),
  stringsAsFactors = FALSE
)
pto_components(comm, tax)
```

`rarefaction_taxonomic` *Taxonomic Diversity Rarefaction*

Description

Computes rarefaction curves for taxonomic diversity indices by subsampling individuals from the community at increasing sample sizes. Uses bootstrap resampling to estimate expected diversity and confidence intervals at each sample size.

Usage

```
rarefaction_taxonomic(
  community,
  tax_tree,
 index = c("shannon", "simpson", "species", "uTO", "TO", "uTO_plus", "TO_plus", "avtd"),
  steps = 20,
  n_boot = 100,
  ci = 0.95,
  seed = NULL,
  correction = c("none", "miller_madow", "grassberger", "chao_shen"),
  parallel = FALSE,
  n_cores = NULL
)
```

Arguments

`community`	A named numeric vector of species abundances (integers). Names must match the first column of `tax_tree`.
`tax_tree`	A data frame with taxonomic hierarchy. First column is species names, subsequent columns are taxonomic ranks.
`index`	Which index to rarefy. One of `"shannon"`, `"simpson"`, `"uTO"`, `"TO"`, `"uTO_plus"`, `"TO_plus"`, `"avtd"`, `"species"` (default: `"shannon"`).
`steps`	Number of sample-size steps along the curve (default: 20).
`n_boot`	Number of bootstrap replicates per step (default: 100).
`ci`	Confidence interval width (default: 0.95).

`seed`	Optional random seed for reproducibility (default: NULL).
`correction`	Bias correction for the Shannon index. One of `"none"` (default), `"miller_madow"`, `"grassberger"`, or `"chao_shen"`. Only used when `index = "shannon"`. Passed to `shannon()`. See `shannon()` for details.
`parallel`	Logical. If `TRUE`, use parallel processing to speed up bootstrap resampling across sample sizes. Default `FALSE`.
`n_cores`	Number of CPU cores to use when `parallel = TRUE`. Default `NULL` uses up to 2 cores (CRAN policy limit).

Details

The algorithm works as follows:

1. Expand the abundance vector into an individual-level vector (e.g., c(sp1=3, sp2=2) becomes c("sp1","sp1","sp1","sp2","sp2")).
2. For each sample size (from min to total N), draw `n_boot` random subsamples without replacement.
3. For each subsample, count species abundances and compute the chosen diversity index.
4. Return the mean and confidence interval at each step.

Value

A data frame with columns:

sample_size Number of individuals in the subsample

mean Mean index value across bootstrap replicates

lower Lower bound of the confidence interval

upper Upper bound of the confidence interval

sd Standard deviation across replicates

References

Gotelli, N.J. & Colwell, R.K. (2001). Quantifying biodiversity: procedures and pitfalls in the measurement and comparison of species richness. Ecology Letters, 4, 379-391.

Ozkan, K. (2018). A new proposed measure for estimating taxonomic diversity. Turkish Journal of Forestry, 19(4), 336-346.

See Also

`plot_rarefaction()` for visualising the rarefaction curve, `ozkan_pto()` for full pTO calculation, `shannon()` and `simpson()` for classical indices, `avtd()` for average taxonomic distinctness.

Examples

```
comm <- c(sp1 = 10, sp2 = 5, sp3 = 8, sp4 = 2, sp5 = 3)
tax <- data.frame(
  Species = paste0("sp", 1:5),
  Genus   = c("G1", "G1", "G2", "G2", "G3"),
  Family  = c("F1", "F1", "F1", "F2", "F2"),
  stringsAsFactors = FALSE
)
rarefaction_taxonomic(comm, tax, index = "shannon", n_boot = 50)
```

`shannon` *Shannon Diversity Index*

Description

Calculates the Shannon-Wiener diversity index (H') for a community, optionally applying a bias correction for small samples.

Usage

```
shannon(
  community,
  base = exp(1),
  correction = c("none", "miller_madow", "grassberger", "chao_shen")
)
```

Arguments

`community`	A numeric vector of species abundances (counts).
`base`	The logarithm base. Default is `exp(1)` (natural log). Use `2` for bits.
`correction`	Bias correction method. One of `"none"` (default, naive MLE), `"miller_madow"`, `"grassberger"`, or `"chao_shen"`. See Details.

Details

The naive (MLE) Shannon index is calculated as:

$$H' = -\sum_{i=1}^{S} p_i \ln(p_i)$$

where $p_i = n_i/N$ is the proportion of species i, N is the total number of individuals, and S is the number of species observed.

The MLE estimator has a known negative bias that is significant for small samples. Three bias-correction methods are available:

Miller-Madow (1955): Adds a first-order bias correction term:

$$H_{MM} = H_{MLE} + \frac{S_{obs} - 1}{2N}$$

Grassberger (2003): Uses the digamma function instead of the logarithm:

$$H_G = \ln(N) - \frac{1}{N} \sum_i n_i \psi(n_i)$$

where ψ is the digamma function.

Chao-Shen (2003): Applies a Good-Turing coverage correction with Horvitz-Thompson weighting:

$$\hat{C} = 1 - f_1/N$$

$$H_{CS} = -\sum_i \frac{\hat{p}_i \ln \hat{p}_i}{1 - (1 - \hat{p}_i)^N}$$

where $\hat{p}_i = \hat{C} \cdot n_i/N$ and f_1 is the number of singletons.

Bias corrections require integer abundance counts. A warning is issued if non-integer values are detected with `correction != "none"`.

Value

A numeric value representing the Shannon diversity index.

References

Miller, G.A. & Madow, W.G. (1954). On the maximum likelihood estimate of the Shannon-Wiener index of diversity. AFCRC-TR-54-75.

Grassberger, P. (2003). Entropy estimates from insufficient samplings. arXiv:physics/0307138.

Chao, A. & Shen, T.-J. (2003). Nonparametric estimation of Shannon's index of diversity when there are unseen species in sample. Environmental and Ecological Statistics, 10, 429-443.

See Also

`simpson()` for Simpson diversity, `deng_entropy_level()` for Deng entropy (a generalization of Shannon).

Examples

```
comm <- c(10, 5, 8, 3, 12)
shannon(comm)
shannon(comm, correction = "miller_madow")
shannon(comm, correction = "grassberger")
shannon(comm, correction = "chao_shen")
```

`simpson` *Simpson Diversity Index*

Description

Calculates the Simpson diversity index (1 - D) for a community.

Usage

```
simpson(community, type = c("gini_simpson", "inverse", "dominance"))
```

Arguments

`community`	A numeric vector of species abundances.
`type`	One of `"inverse"` (1/D), `"gini_simpson"` (1 - D), or `"dominance"` (D). Default is `"gini_simpson"`.

Details

Simpson's dominance index D is calculated as:

$$D = \sum_{i=1}^{S} p_i^2$$

The Gini-Simpson index is $1 - D$ and the inverse Simpson is $1/D$.

Value

A numeric value representing the Simpson index.

See Also

`shannon()` for Shannon diversity.

Examples

```
comm <- c(10, 5, 8, 3, 12)
simpson(comm)
simpson(comm, type = "inverse")
```

`simulate_td` *Simulate Expected AvTD/VarTD Under Random Sampling*

Description

Generates the null distribution of Average Taxonomic Distinctness (AvTD) and/or Variation in Taxonomic Distinctness (VarTD) by randomly drawing species subsets from a regional species pool. Used to construct funnel plots for statistical testing (Clarke & Warwick 1998, 2001).

Usage

```
simulate_td(
  tax_tree,
  s_range = NULL,
  n_sim = 999L,
  index = c("both", "avtd", "vartd"),
  weights = NULL,
  ci = 0.95,
  seed = NULL,
  parallel = FALSE,
  n_cores = NULL
)
```

Arguments

`tax_tree`	A data frame representing the full regional species pool taxonomy. First column is species names, subsequent columns are taxonomic ranks from lowest to highest.
`s_range`	Integer vector of species richness values to simulate. Default `NULL` uses `2:S` where S is the total number of species in `tax_tree`.
`n_sim`	Number of random draws per species richness value (default 999).
`index`	Which index to simulate: `"avtd"`, `"vartd"`, or `"both"` (default).
`weights`	Optional numeric vector of weights for taxonomic levels. Passed to `avtd()` and `vartd()`.
`ci`	Confidence interval width (default 0.95).
`seed`	Optional random seed for reproducibility.
`parallel`	Logical. If `TRUE`, use parallel processing to speed up simulations. Default `FALSE`.
`n_cores`	Number of CPU cores to use when `parallel = TRUE`. Default `NULL` uses up to 2 cores (CRAN policy limit).

Details

For each value of S in `s_range`, `n_sim` random subsets of S species are drawn (without replacement) from the full species pool in `tax_tree`. AvTD and/or VarTD are computed for each random subset. The mean and percentile-based confidence limits are recorded.

The resulting object can be passed to `plot_funnel()` to produce the classic Clarke & Warwick funnel plot.

Value

A data frame with class `"td_simulation"` containing columns:

s Species richness (number of species drawn)

mean_avtd Mean simulated AvTD (if index includes avtd)

lower_avtd Lower CI bound for AvTD

upper_avtd Upper CI bound for AvTD

mean_vartd Mean simulated VarTD (if index includes vartd)

lower_vartd Lower CI bound for VarTD

upper_vartd Upper CI bound for VarTD

Attributes: `ci`, `index`, `n_sim`, `pool_size`.

References

Clarke, K.R. & Warwick, R.M. (1998). A taxonomic distinctness index and its statistical properties. Journal of Applied Ecology, 35, 523-531.

Clarke, K.R. & Warwick, R.M. (2001). A further biodiversity index applicable to species lists: variation in taxonomic distinctness. Marine Ecology Progress Series, 216, 265-278.

See Also

`plot_funnel()` for visualisation, `avtd()` and `vartd()` for the underlying calculations.

Examples

```
tax <- data.frame(
  Species = paste0("sp", 1:10),
  Genus   = rep(c("G1", "G2", "G3", "G4", "G5"), each = 2),
  Family  = rep(c("F1", "F1", "F2", "F2", "F3"), each = 2),
  Order   = rep(c("O1", "O1", "O2", "O2", "O2"), each = 2),
  stringsAsFactors = FALSE
)
sim <- simulate_td(tax, n_sim = 99, seed = 42)
sim
```

`tax_distance_matrix` *Compute Taxonomic Distance Matrix*

Description

Calculates pairwise taxonomic distances between species based on their positions in a taxonomic hierarchy. Distance is computed as the weighted proportion of taxonomic levels at which two species diverge.

Usage

```
tax_distance_matrix(tax_tree, species = NULL, weights = NULL)
```

Arguments

`tax_tree`	A data frame representing the taxonomic hierarchy. First column must be species names, subsequent columns are taxonomic ranks from lowest to highest.
`species`	Optional character vector of species names to include. If NULL, all species in `tax_tree` are used.
`weights`	Optional numeric vector of weights for each taxonomic level. If NULL, equal weights are assigned.

Value

A symmetric matrix of taxonomic distances between species. With default equal step weights (1, 2, 3, ...), values range from 0 (same species) to the number of taxonomic levels (maximum distance when no common ancestor is found at any level).

Examples

```
tax <- data.frame(
  Species = c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis"),
  Genus = c("Quercus", "Pinus", "Fagus"),
  Family = c("Fagaceae", "Pinaceae", "Fagaceae"),
  Order = c("Fagales", "Pinales", "Fagales"),
  stringsAsFactors = FALSE
)

tax_distance_matrix(tax)
```

`vartd` *Variation in Taxonomic Distinctness (Lambda+)*

Description

Calculates the variation in taxonomic distinctness (VarTD, Lambda+) based on Clarke & Warwick (2001).

Usage

```
vartd(species, tax_tree, weights = NULL)
```

Arguments

`species`	Character vector of species names present in the community.
`tax_tree`	A data frame representing the taxonomic hierarchy.
`weights`	Optional numeric vector of weights for taxonomic levels.

Value

A numeric value representing the variation in taxonomic distinctness (Lambda+).

References

Clarke, K.R. & Warwick, R.M. (2001). A further biodiversity index applicable to species lists: variation in taxonomic distinctness. Marine Ecology Progress Series, 216, 265-278.

Examples

```
tax <- data.frame(
  Species = c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis",
              "Abies_nordmanniana"),
  Genus = c("Quercus", "Pinus", "Fagus", "Abies"),
  Family = c("Fagaceae", "Pinaceae", "Fagaceae", "Pinaceae"),
  Order = c("Fagales", "Pinales", "Fagales", "Pinales"),
  stringsAsFactors = FALSE
)

spp <- c("Quercus_robur", "Pinus_nigra", "Fagus_orientalis")
vartd(spp, tax)
```

Index