Package: b3gbi (via r-universe)

October 21, 2026

Type Package

Title General Biodiversity Indicators for Biodiversity Data Cubes

Version 1.1.3

Description Calculate general biodiversity indicators from species occurrence cubes, such as those produced by the 'GBIF' (Global Biodiversity Information Facility) occurrence cube service <https://techdocs.gbif.org/en/data-use/data-cubes>. Indicators include observed and coverage-based rarefied species richness and Hill numbers (via 'iNEXT'), evenness, rarity, taxonomic distinctness, occupancy, turnover and sample completeness, and can be calculated over time (time series) or space (maps), with bootstrap confidence intervals. Cubes using the EEA reference grid, MGRS, the extended quarter-degree grid (EQDGC) or ISEA3H grids are supported. For details see Dove (2026) <doi:10.21105/joss.10963>.

License MIT + file LICENSE

URL https://github.com/b-cubed-eu/b3gbi, https://b-cubed-eu.github.io/b3gbi/

Depends R (>= 3.5.0)

Imports boot, dplyr, ggplot2, iNEXT, magrittr, purrr, readr, rlang, rnaturalearth, scales, sf, stringr, tibble, tidyr, tidyselect, units

Suggests bit64, dggridR, dubicube, knitr, mapview, mockery, mockr, patchwork, RColorBrewer, rgbif (>= 3.7.0), rmarkdown, rnaturalearthdata, rnaturalearthhires, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Config/Needs/check b-cubed-eu/dubicube, ropensci/rnaturalearthhires

Config/Needs/coverage b-cubed-eu/dubicube, ropensci/rnaturalearthhires

Config/Needs/website b-cubed-eu/dubicube, ropensci/rnaturalearthhires

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

VignetteBuilder knitr, rmarkdown

Additional_repositories https://b-cubed-eu.r-universe.dev, https://ropensci.r-universe.dev

BugReports https://github.com/b-cubed-eu/b3gbi/issues

NeedsCompilation no

Author Shawn Dove [aut, cre] (ORCID: <https://orcid.org/0000-0001-9465-5638>), Ward Langeraert [ctb] (ORCID: <https://orcid.org/0000-0002-5900-8109>), Peter Desmet [ctb] (ORCID: <https://orcid.org/0000-0002-8442-8025>), University of Giessen [cph], European Union [fnd] (Horizon Europe Research and Innovation Programme, grant 101059592)

Maintainer Shawn Dove <Shawn.Dove@allzool.bio.uni-giessen.de>

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

Date/Publication 2026-10-21 15:40:19 UTC

RemoteUrl https://github.com/cran/b3gbi

RemoteRef HEAD

RemoteSha b2bebcdbb40a1bec77f0360edee69606a1fa261a

Contents

add_ci . . . 3
area_rarity_map . . . 7
available_indicators . . . 11
calc_ci . . . 12
calc_map.completeness . . . 14
calc_ts.default . . . 16
completeness_map . . . 19
compute_indicator_workflow . . . 22
convert_eqdgc_latlong . . . 25
create_sf_from_utm . . . 26
cum_richness_ts . . . 27
example_cube_1 . . . 30
example_indicator_map1 . . . 31
example_indicator_ts1 . . . 32
example_indicator_ts2 . . . 32
get_observed_years . . . 33
hill0_map . . . 33
list_species . . . 39
mgrs_to_latlong_bbox . . . 39
newness_map . . . 40
obs_richness_map . . . 43

occ_density_map 46
occ_turnover_ts 49
pielou_evenness_map 51
plot_map 55
plot_mv 59
plot_species_map 60
plot_species_ts 64
plot_ts 67
prepare_indicator_bootstrap 70
print.available_indicators 72
print.indicator_map 72
print.indicator_ts 73
print.processed_cube 74
print.processed_cube_dsinfo 74
print.sim_cube 75
process_cube 75
relative_occupancy_map 78
replace_na 82
spec_occ_map 83
spec_range_map 86
spec_richness_density_map 88
sum_by_taxon 91
tax_distinct_map 92
total_occ_map 95

Index **99**

`add_ci` *Add Confidence Intervals to an Indicator Object*

Description

This function calculates bootstrap confidence intervals for an existing `indicator_ts` object. It supports both cube-level bootstrapping (resampling occurrence records) and indicator-level bootstrapping (resampling the per-year components of the indicator), allowing for advanced transformations during the CI calculation process.

Usage

```
add_ci(
  indicator,
  num_bootstrap = 1000,
  bootstrap_level = c("auto", "cube", "indicator"),
  ci_type = c("perc", "bca", "norm", "basic", "none"),
  trans = function(t) t,
  inv_trans = function(t) t,
  confidence_level = 0.95,
  overwrite = TRUE,
```

```
  boot_args = list(),
  ci_args = list(),
  seed = 123,
  ...
)
```

Arguments

`indicator` An object of class `indicator_ts` to which confidence intervals should be added.

`num_bootstrap` (Optional) Number of bootstrap replicates to perform. (Default: 1000)

`bootstrap_level`

(Optional) Level at which to perform bootstrapping:

- `auto` (default): Uses `cube` if the **dubicube** package is installed, and otherwise falls back to `indicator` (with a message).
- `cube`: Bootstrapping is done by resampling the occurrence records in the cube, using the **dubicube** package. This is statistically more robust as it captures the underlying sampling uncertainty. Requires **dubicube**.
- `indicator`: Within each year, the component values of the indicator (occurrence records, species, or grid-cell values) are resampled with **boot**, without recalculating the indicator from a resampled cube. This is faster for large cubes but less robust, and does not require **dubicube**.

The level that was used is stored in the `ci_method` element of the returned object and shown when it is printed.

`ci_type` (Optional) Type of bootstrap confidence intervals to calculate. (Default: `"perc"`). Supported options are:

- `perc`: Percentile intervals.
- `bca`: Bias-corrected and accelerated intervals.
- `norm`: Normal approximation intervals.
- `basic`: Basic bootstrap intervals.
- `none`: No confidence intervals calculated; the indicator is returned unchanged (with a message).

`trans` (Optional) A function for transforming the indicator values before calculating confidence intervals (e.g., `log`). At indicator level it is passed to `boot::boot.ci()` as `h` (not used for Hill numbers). The `trans`/`inv_trans` arguments are ignored for evenness indicators at cube level, which always use the logit transformation unless overridden via `boot_args` or `ci_args`. (Default: identity function)

`inv_trans` (Optional) The inverse of the transformation function `trans` (e.g., `exp`). Used to back-transform the intervals to the original scale. At indicator level it is passed to `boot::boot.ci()` as `hinv`. (Default: identity function)

`confidence_level`

(Optional) The confidence level for the calculated intervals (e.g., 0.95 for 95% CIs). Used at both bootstrap levels. (Default: 0.95)

`overwrite` (Optional) Logical. If the indicator already contains confidence intervals (`ll` and `ul` columns), should they be replaced? (Default: TRUE)

`boot_args`	(Optional) Named list of additional arguments passed to `dubicube::bootstrap_cube()` (cube level only). (Default: `list()`)
`ci_args`	(Optional) Named list of additional arguments passed to `dubicube::calculate_bootstrap_ci()` (cube level only). (Default: `list()`)
`seed`	(Optional) Integer. Random seed for bootstrapping, used at both bootstrap levels. The random number generator state of the session is restored afterwards. Use `NA` to not set a seed. (Default: 123)
`...`	(Optional) Additional arguments passed to `calc_ci()` (indicator level only).

Details

For cube-level bootstrapping, the function acts as a bridge to the **dubicube** package (Langeraert et al.), which is developed alongside **b3gbi** within the B-Cubed project. **dubicube** is not on CRAN; it can be installed from R-universe with `install.packages("dubicube", repos = c("https://b-cubed-eu.r-universe.dev", "https://cloud.r-project.org"))`.

Indicator-specific defaults:

Depending on the indicator, default settings are internally applied when calculating bootstrap confidence intervals. These defaults control whether bootstrapping is performed per group, which transformation is used, and whether bias correction is disabled.

The following defaults are used unless explicitly overridden via `trans`, `inv_trans`, `boot_args`, or `ci_args`:

- `total_occ`
  - Group-specific bootstrapping: **yes**
  - Transformation: **none (identity)**
  - Bias correction: **disabled** (`no_bias = TRUE`)
- `spec_occ`, `spec_range`
  - Group-specific bootstrapping: **no** (whole-cube resampling, with intervals per year and species)
  - Transformation: **none (identity)**
  - Bias correction: enabled
- `pielou_evenness`, `williams_evenness`
  - Group-specific bootstrapping: **no**
  - Transformation: **logit**
  - Inverse transformation: **inverse logit**
  - Bias correction: enabled
- `occ_density`, `ab_rarity`, `area_rarity`, `newness`
  - Group-specific bootstrapping: **no**
  - Transformation: **none (identity)**
  - Bias correction: enabled

Group-specific bootstrapping means that resampling is performed within each group (e.g., year), which is required for indicators that are inherently group-based. This is in contrast to whole-cube bootstrapping, where resampling is performed across the whole dataset; this is applicable to indicators that combine information across groups.

Transformations are applied prior to confidence interval calculation and inverted afterwards to return intervals on the original scale.

These defaults (grouping, the logit transformation for evenness and the disabled bias correction for `total_occ`) apply to cube-level bootstrapping. At indicator level, `trans` and `inv_trans` are passed to `boot::boot.ci()` for all indicators (including evenness) except Hill numbers.

Indicators outside scope of this function:

For certain indicators, confidence intervals cannot be calculated or added post-hoc because they are not statistically relevant, or because proper uncertainty estimation is not supported. In such cases, a warning is issued and the original object is returned. The following indicators cannot have confidence intervals added via `add_ci()`:

- `obs_richness`, `spec_richness_density`: Observed species richness and richness density are highly sensitive to sampling effort and sample size. Furthermore, bootstrapping observed occurrences can never discover new/unobserved species, meaning resampled richness estimates are always less than or equal to the observed richness. This results in confidence intervals that lie entirely at or below the observed value (i.e., the upper bound cannot exceed the observed richness). Therefore, post-hoc bootstrapping is not a statistically sound way to estimate their uncertainty (Hill numbers `hill0`, `hill1`, `hill2` should be used instead for proper rarefaction/extrapolation).
- `completeness`: Sample completeness (coverage) is calculated as a deterministic sample statistic using `iNEXT` methods, and confidence interval calculation is not supported.
- `relative_occupancy`: Standard bootstrapping is not supported because resampling strips spatial attributes (like `total_num_cells` or `total_area_sqkm`) necessary for occupancy calculation. These spatial attributes cannot be regenerated from the resampled occurrences alone, as occurrences only capture cells/regions where species were detected and contain no information about empty grid cells (zero occurrences) in the study area.
- `cum_richness` (Cumulative Species Richness): This is an inherently temporal and sequential indicator for which CIs are not generated, as its nature as an accumulating count over time is not well-suited to standard bootstrapping methods for uncertainty.
- `occ_turnover` (Species Turnover): Confidence intervals are not calculated for this indicator. The metric's reliance on comparing the unique species lists between consecutive time steps means that bootstrapping individual observations within each time step would introduce artificial variability in these lists. This makes the resulting gains and losses highly unstable and would lead to unreliable or biased confidence intervals for the turnover metric.
- `tax_distinct` (Taxonomic Distinctness): Confidence intervals are not calculated for this indicator. Due to its high sensitivity to the exact species composition and taxonomic relationships within a sample, bootstrapping individual occurrences can introduce significant artificial variability, leading to unreliable or biased confidence intervals.

Note: For Hill numbers (`hill0`, `hill1`, `hill2`), cube-level bootstrapping is not supported natively, but `add_ci()` will automatically switch to indicator-level bootstrapping and calculate confidence intervals internally using the `iNEXT` package.

Value

The input `indicator_ts` object, with the bootstrap level used stored in its `ci_method` element (`"cube"` or `"indicator"`) and the following columns added to its `data`:

- `ll`: Lower limit of the confidence interval (negative lower limits are set to 0).

- `ul`: Upper limit of the confidence interval.

Except for Hill numbers (whose intervals come from **iNEXT**), the following columns are also added:

- `int_type`: The type of interval calculated (e.g., `"perc"`).
- `conf`: The confidence level used.
- `est_boot`: The bootstrap estimate of the indicator value.
- `se_boot`: The bootstrap standard error.
- `bias_boot`: The bootstrap estimate of bias.

At cube level, `est_boot`, `se_boot` and `bias_boot` are only returned for some indicators (e.g., `total_occ`, `spec_occ` and `spec_range`), depending on the bootstrap method and bias correction used.

If `ci_type = "none"`, or if confidence intervals cannot be calculated for the indicator, the input object is returned unchanged.

See Also

`dubicube::bootstrap_cube()`, `dubicube::calculate_bootstrap_ci()`

Examples

```
# Load sample cube data
cube_path <- system.file("extdata", "denmark_mammals_cube_eea.csv", package = "b3gbi")
cube <- process_cube(cube_path)

# Calculate a time series indicator
ts_occ <- total_occ_ts(cube)

# Add bootstrap confidence intervals
ts_occ_ci <- add_ci(ts_occ, num_bootstrap = 100)
plot(ts_occ_ci)
```

`area_rarity_map` *Calculate Rarity Over Time or Space*

Description

This function calculates area-based or abundance-based rarity over a gridded map or as a time series (see 'Details' for more information).

Usage

```
area_rarity_map(data, ...)

area_rarity_ts(data, ...)

ab_rarity_map(data, ...)

ab_rarity_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Rarity:

Rarity is the scarcity or infrequency of a particular species in an area. A rare species might have a small population size, a limited distribution, or a unique ecological niche (Maciel, 2021; Rabinowitz, 1981). Rarity can also be a biodiversity indicator when summed over multiple species in an area, and may provide important insight for determining conservation priorities. When measured over time, rarity may indicate potential threats or changes in the environment.

Abundance-Based Rarity:

Abundance-based rarity is the inverse of the proportion of total occurrences represented by a particular species. The total summed rarity for each grid cell or year is calculated (sum the rarity values of each species present there). It is calculated as:

$$\sum_{i=1}^{S} \frac{1}{p_i}$$

where S is the number of species and pi is the proportion of occurrences represented by species i. For maps, pi is calculated within each grid cell (over the whole time period). For time series, pi is calculated separately for each year (species i's occurrences that year divided by all occurrences that year), and each species present in a year contributes once.

Area-Based Rarity:

Area-based rarity is the inverse of occupancy frequency (the proportion of occupied grid cells in which the species occurs) for a particular species. The total summed rarity for each grid cell or year is calculated (sum the rarity values of each species present there). It is calculated as:

$$\sum_{i=1}^{S} \frac{N}{n_i}$$

where S is the number of species, N is the total number of occupied grid cells, and ni is the number of grid cells occupied by species i.

For maps, N and ni are calculated over the whole time period, and the summed rarity is reported for each grid cell. For time series, N and ni are calculated separately for each year; rarity is summed within each occupied grid cell and the yearly value is the mean of these cell sums.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'area_rarity' or 'ab_rarity' containing the calculated indicator values and metadata.

Functions

- `area_rarity_map()`:
- `area_rarity_ts()`:
- `ab_rarity_map()`:
- `ab_rarity_ts()`:

References

Maciel, E. A. (2021). An index for assessing the rare species of a community. *Ecological Indicators*, 124, 107424.

Rabinowitz, D. (1981). Seven forms of rarity. *Biological aspects of rare * *plant conservation*.

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
arr_map <- area_rarity_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(arr_map)


arr_ts <- area_rarity_ts(example_cube_1, first_year = 1985)
plot(arr_ts)


abr_map <- ab_rarity_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(abr_map)


abr_ts <- ab_rarity_ts(example_cube_1, first_year = 1985)
plot(abr_ts)
```

`available_indicators` *Indicators Available for Use in the Package*

Description

A list of all biodiversity indicators available within the package, along with the dimensions they can be calculated across, the functions to access them, and any special arguments.

Usage

```
available_indicators
```

Format

A special object of class 'available_indicators': a named list of 19 indicators, each a list of nine fields with information about it

indicator_class class of the indicator

indicator_name name of the indicator

plot_title title to be used when plotting with automated title generation

legend_label title to be used when plotting with automated legend title generation

legend_transformation any transformation to perform on the legend when plotting, to improve visualization of maps

map_wrapper wrapper function to use when calculating indicator as a map

ts_wrapper wrapper function to use when calculating indicator as a time series

map_function_arguments any special arguments to consider when using the function to calculate an indicator map

ts_function_arguments any special arguments to consider when using the function to calculate an indicator time series

Source

N/A

`calc_ci` *Calculate Confidence Intervals for a Biodiversity Indicator*

Description

Calculates indicator-level bootstrap confidence intervals. It is called internally by `add_ci()` (indicator level) or when `ci_type` is supplied to a `*_ts()` function; it is not meant to be called directly.

Usage

```
calc_ci(x, indicator, ...)

## Default S3 method:
calc_ci(x, indicator, ...)

## S3 method for class 'total_occ'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'occ_density'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'spec_richness_density'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'newness'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'williams_evenness'
calc_ci(x, ...)

## S3 method for class 'pielou_evenness'
calc_ci(x, ...)

## S3 method for class 'ab_rarity'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)
```

```
## S3 method for class 'area_rarity'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'spec_occ'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'spec_range'
calc_ci(x, indicator, num_bootstrap = 1000, ci_type = "perc", ...)

## S3 method for class 'hill0'
calc_ci(x, indicator, num_bootstrap = 1000, ...)

## S3 method for class 'hill1'
calc_ci(x, indicator, num_bootstrap = 1000, ...)

## S3 method for class 'hill2'
calc_ci(x, indicator, num_bootstrap = 1000, ...)
```

Arguments

`x`	A data cube object
`indicator`	An indicator calculated over time, in the form of a data frame. *Note: this should NOT be an 'indicator_ts' object as it is meant to be called by the 'compute_indicator_workflow' function.
`...`	Additional arguments passed to specific calc_ci functions and on to `boot::boot.ci()` (e.g., `conf`, `h`, `hinv`), or to the **iNEXT**-based calculation for Hill numbers (e.g., `conf`).
`num_bootstrap`	(Optional) Set the number of bootstraps to calculate for generating confidence intervals. (Default: 1000)
`ci_type`	(Optional) Type of bootstrap interval passed to `boot::boot.ci()`. (Default: "perc")

Value

A data frame containing indicator values with calculated lower (`ll`) and upper (`ul`) confidence bounds.

Methods (by class)

- `calc_ci(total_occ)`: Calculate confidence intervals for total occurrences
- `calc_ci(occ_density)`: Calculate confidence intervals for occurrence density
- `calc_ci(spec_richness_density)`: Calculate confidence intervals for species richness density
- `calc_ci(newness)`: Calculate confidence intervals for newness
- `calc_ci(williams_evenness)`: Calculate confidence intervals for Williams' evenness
- `calc_ci(pielou_evenness)`: Calculate confidence intervals for Pielou's evenness

- `calc_ci(ab_rarity)`: Calculate confidence intervals for abundance-based rarity
- `calc_ci(area_rarity)`: Calculate confidence intervals for area-based rarity
- `calc_ci(spec_occ)`: Calculate confidence intervals for species occurrences
- `calc_ci(spec_range)`: Calculate confidence intervals for species range
- `calc_ci(hill0)`: Calculate confidence intervals for Hill0 (Richness)
- `calc_ci(hill1)`: Calculate confidence intervals for Hill1 (Shannon)
- `calc_ci(hill2)`: Calculate confidence intervals for Hill2 (Simpson)

Examples

```
# calc_ci() is called automatically when confidence intervals are requested
# during indicator calculation (or by add_ci(bootstrap_level = "indicator"))
occ_ts <- total_occ_ts(example_cube_1, first_year = 2000,
                       ci_type = "perc", num_bootstrap = 100)
head(occ_ts$data)
```

`calc_map.completeness` *Calculate Biodiversity Indicators Over Space*

Description

This function provides a flexible framework for calculating various biodiversity indicators on a spatial grid or as a time series. It prepares the data, creates a grid, calculates indicators, and formats the output into an appropriate S3 object ('indicator_map'). Specific implementations for different indicator types are provided using the appropriate wrappers.

Usage

```
## S3 method for class 'completeness'
calc_map(x, ...)

## S3 method for class 'hill0'
calc_map(x, ...)

## S3 method for class 'hill1'
calc_map(x, ...)

## S3 method for class 'hill2'
calc_map(x, ...)

## S3 method for class 'relative_occupancy'
calc_map(x, occ_type = 0, ...)

## S3 method for class 'obs_richness'
calc_map(x, ...)
```

```
## S3 method for class 'total_occ'
calc_map(x, ...)

## S3 method for class 'newness'
calc_map(x, newness_min_year = NULL, ...)

## S3 method for class 'spec_richness_density'
calc_map(x, ...)

## S3 method for class 'occ_density'
calc_map(x, ...)

## S3 method for class 'williams_evenness'
calc_map(x, ...)

## S3 method for class 'pielou_evenness'
calc_map(x, ...)

## S3 method for class 'ab_rarity'
calc_map(x, ...)

## S3 method for class 'area_rarity'
calc_map(x, ...)

## S3 method for class 'spec_occ'
calc_map(x, ...)

## S3 method for class 'spec_range'
calc_map(x, ...)

## S3 method for class 'tax_distinct'
calc_map(x, ...)

calc_map(x, ...)
```

Arguments

`x` A data cube object ('processed_cube').

`...` Additional arguments passed to specific indicator calculation functions.

`occ_type` Integer controlling the occupancy denominator (default `0`):

- `0` — **Total-area occupancy**: number of grid cells occupied by the species (across all years) / total number of grid cells in the study region (including unsampled cells). *Presence-only caveat*: empty cells cannot be assumed truly unoccupied; they may simply be unsampled.
- `1` — **Relative-to-ever-occupied occupancy**: species' cells / cells with *at least one occurrence (any species, any year)*. Conditions on cells where sampling effort is documented.

- 2 — **Temporal mean annual occupancy**: for each year, compute the proportion of that year's occupied cells (any species) in which the species was recorded; then average those annual proportions across the years in which the species was recorded (years without records of the species do not contribute a zero). This captures how consistently a species occupies the active sampling footprint over time.

Note on presence-only data: All three types rely on presence-only records. A cell with no records cannot be assumed to be truly unoccupied. Types 1 and 2 condition on cells with documented occurrences, but those still reflect sampling effort rather than true species absence.

Note on cell aggregation: When a coarser `cell_size` is chosen, data are aggregated to coarser grid cells before this calculation. All denominators count post-aggregation cells (`cellid`).

`newness_min_year`

(Optional) If set, only shows values above this (e.g. 1970). Values below the minimum will be replaced with NA. This can be useful e.g. if you have outlier cells where the data is very old causing the legend gradient to stretch in a way that makes other cell values difficult to discern.

Value

An S3 object of the class 'indicator_map' containing the calculated indicator values and metadata.

Examples

```
observed_richness_map <- obs_richness_map(example_cube_1, level = "country",
                                          region = "Denmark")
plot(observed_richness_map)
```

`calc_ts.default` *Calculate Biodiversity Indicators Over Time*

Description

This function provides a flexible framework for calculating various biodiversity indicators over time. It prepares the data, creates a grid, calculates indicators, and formats the output into an appropriate S3 object ('indicator_ts'). Specific implementations for different indicator types are provided using the appropriate wrappers.

Usage

```
## Default S3 method:
calc_ts(x, ...)

## S3 method for class 'completeness'
```

```
calc_ts(x, ...)

## S3 method for class 'hill0'
calc_ts(x, ...)

## S3 method for class 'hill1'
calc_ts(x, ...)

## S3 method for class 'hill2'
calc_ts(x, ...)

## S3 method for class 'obs_richness'
calc_ts(x, ...)

## S3 method for class 'cum_richness'
calc_ts(x, ...)

## S3 method for class 'total_occ'
calc_ts(x, ...)

## S3 method for class 'spec_richness_density'
calc_ts(x, ...)

## S3 method for class 'occ_density'
calc_ts(x, ...)

## S3 method for class 'newness'
calc_ts(x, ...)

## S3 method for class 'williams_evenness'
calc_ts(x, ...)

## S3 method for class 'pielou_evenness'
calc_ts(x, ...)

## S3 method for class 'ab_rarity'
calc_ts(x, ...)

## S3 method for class 'area_rarity'
calc_ts(x, ...)

## S3 method for class 'spec_occ'
calc_ts(x, ...)

## S3 method for class 'spec_range'
calc_ts(x, ...)

## S3 method for class 'relative_occupancy'
```

```
calc_ts(x, occ_type = 0, ...)

## S3 method for class 'tax_distinct'
calc_ts(x, set_rows = 1, ...)

## S3 method for class 'occ_turnover'
calc_ts(x, ...)

calc_ts(x, ...)
```

Arguments

`x` A data cube object ('processed_cube').

`...` Additional arguments passed to specific indicator calculation functions.

`occ_type` Integer controlling the occupancy denominator (default `0`):

- `0` — **Total-area occupancy**: number of cells occupied by species / total number of grid cells in the study region (including unsampled cells). This is the most conservative measure. *Presence-only caveat*: empty cells cannot be assumed to be truly unoccupied; they may simply lack sampling effort.
- `1` — **Relative-to-ever-occupied occupancy**: number of cells occupied by species / number of cells with *at least one occurrence (any species) anywhere in the time window*. This conditions on cells where some sampling effort is documented but the denominator is still constant across years.
- `2` — **Annual occupancy**: for each year, number of cells occupied by species / number of cells with at least one occurrence (any species) *in that year*. The denominator therefore varies by year, reflecting changes in sampling footprint over time.

Note on presence-only data: All three types are computed from presence-only occurrence data. A cell with no records cannot be assumed to be truly unoccupied; it may be unsampled or under-surveyed. Types 1 and 2 partially address this by restricting the denominator to cells with documented occurrences, but they still reflect *recording effort* rather than true absence. Interpret all types with care.

Note on cell aggregation: When a coarser `cell_size` is chosen, data are aggregated to coarser grid cells before this calculation. The denominator counts post-aggregation cells (`cellid`), not original cube cells (`cellCode`).

`set_rows` Deprecated and ignored. Taxa are now looked up by their GBIF taxon key, so there is no ambiguity to resolve.

Value

An S3 object of the class 'indicator_ts' containing the calculated indicator values and metadata.

Examples

```
total_occurrences_trend <- total_occ_ts(example_cube_1)
plot(total_occurrences_trend, min_year=1980)
```

completeness_map *Calculate Completeness (Sample Coverage) Over Space or Time*

Description

This function calculates the completeness of biodiversity data over a gridded map or as a time series using the Sample Coverage metric from the iNEXT package.

Usage

```
completeness_map(
  data,
  cutoff_length = 5,
  data_type = c("incidence", "abundance"),
  assume_freq = FALSE,
  ...
)

completeness_ts(data, cutoff_length = 5, gridded_average = FALSE, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`cutoff_length` (Optional) Minimum amount of data required for a grid cell or year to be included. For maps, grid cells with fewer than `cutoff_length` species are removed. For time series, years whose species-by-cell incidence matrix has `cutoff_length` entries or fewer are removed (with `gridded_average = TRUE`: coarse grid cells containing `cutoff_length` or fewer native cells with records that year). Default is 5.

`data_type` (Optional, maps only) The type of data: "incidence" or "abundance". Default is "incidence".

`assume_freq` (Optional, maps only) Whether to assume frequency data if using incidence. Default is FALSE.

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km,

and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.

- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around

the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

`gridded_average`

(Optional, time series only) If TRUE, completeness is calculated separately for each cell of a grid coarser than the cube (using the native cube cells within it as sampling units), and the yearly value is the mean over these grid cells, rather than calculating it for the entire area at once. If `cell_size` is left at its default ("grid"), a grid 4 times coarser than the cube's resolution is used; otherwise `cell_size` must be coarser than the cube's resolution. Default is FALSE.

Details

Completeness (Sample Coverage):

Completeness is measured as **Sample Coverage**, a concept developed by Good (1953), crediting Turing, and further popularized in ecology by Chao and Jost (2012). Sample coverage estimates the proportion of the total individuals in an ecological community that belong to the species detected in a sample.

A coverage value of 1.0 indicates that no new species are expected to be uncovered through further sampling, meaning the sample is perfectly complete. A value of 0.8, for example, suggests that 20% of the individuals in the community belong to species that have not yet been detected in the sample.

Calculating sample coverage is a prerequisite for reliable comparisons of biodiversity across different areas or time periods, as it provides a standardized measure of sample completeness that is independent of sample size alone (Chao et al., 2014).

In this package, completeness is calculated using the 'iNEXT' package based on the observed data in each grid cell or time point. For time series, occurrences are converted to incidence data with the native grid cells of the cube as sampling units. For maps, see `data_type` and `assume_freq` (by default, years are the sampling units).

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'completeness' containing the calculated indicator values and metadata.

Functions

- `completeness_map()`:

- completeness_ts():

References

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, *40*(3-4), 237-264.

Chao, A., & Jost, L. (2012). Coverage-based rarefaction and extrapolation: standardizing samples by completeness rather than size. *Ecology*, *93*(12), 2533-2547.

Chao, A., Gotelli, N. J., Hsieh, T. C., Sander, E. L., Ma, K. H., Colwell, R. K., & Ellison, A. M. (2014). Rarefaction and extrapolation with Hill numbers: a framework for sampling and estimation in species diversity studies. *Ecological monographs*, *84*(1), 45-67.

See Also

compute_indicator_workflow()

Examples

```
comp_map <- completeness_map(example_cube_1)
plot(comp_map)


comp_ts <- completeness_ts(example_cube_1, first_year = 1985)
plot(comp_ts)
```

---

compute_indicator_workflow

Calculate Biodiversity Indicators Over Space or Time

Description

This function provides a flexible framework for calculating various biodiversity indicators on a spatial grid or as a time series. It prepares the data, creates a grid, calculates indicators, and formats the output into an appropriate S3 object ('indicator_map' or 'indicator_ts').

Usage

```
compute_indicator_workflow(
  data,
  type,
  dim_type = c("map", "ts"),
  cell_size = "grid",
  level = c("cube", "continent", "country", "world", "sovereignty", "geounit"),
  region = "Europe",
  ne_type = c("countries", "map_units", "sovereignty", "tiny_countries"),
  ne_scale = c("medium", "small", "large"),
```

```
  output_crs = NULL,
  first_year = NULL,
  last_year = NULL,
  spherical_geometry = TRUE,
  make_valid = FALSE,
  shapefile_path = NULL,
  shapefile_crs = NULL,
  invert = FALSE,
  include_land = TRUE,
  include_ocean = TRUE,
  buffer_dist_km = 50,
  force_grid = FALSE,
  ...
)
```

Arguments

`data` A data cube object (class 'processed_cube', 'processed_cube_dsinfo' or 'sim_cube').

`type` The indicator to calculate. Supported options include:

- 'obs_richness': Observed species richness.
- 'total_occ': Total number of occurrences.
- 'newness': Mean year of occurrence.
- 'occ_density': Density of occurrences.
- 'spec_richness_density': Species richness density (richness / area).
- 'williams_evenness', 'pielou_evenness': Evenness measures.
- 'ab_rarity', 'area_rarity': Abundance-based and area-based rarity scores.
- 'cum_richness': Cumulative species richness.
- 'occ_turnover': Occupancy turnover.
- 'spec_range': Species range size.
- 'spec_occ': Species occurrences.
- 'relative_occupancy': Species relative occupancy.
- 'tax_distinct': Taxonomic distinctness.
- 'hill0': Species richness (estimated by coverage-based rarefaction).
- 'hill1': Hill-Shannon diversity (estimated by coverage-based rarefaction).
- 'hill2': Hill-Simpson diversity (estimated by coverage-based rarefaction).
- 'completeness': Sample completeness (Sample Coverage).

`dim_type` (Optional) Dimension to calculate indicator over time: 'ts', or space: 'map'. (Default: 'map')

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.

- "auto": determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as "10km" or "500m".

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

`...` Additional arguments passed to specific indicator calculation functions. For time series, `ci_type` (default `"none"`) and `num_bootstrap` (default 0) can be used to request bootstrapped confidence intervals directly (alternatively, use `add_ci()` afterwards). Other examples are `newness_min_year` for `newness_map()` and `occ_type` for `relative_occupancy_map()` and `relative_occupancy_ts()`.

Value

An object of class "indicator_map" (dim_type = "map") or "indicator_ts" (dim_type = "ts") containing the calculated indicator values and metadata.

Examples

```
diversity_map <- compute_indicator_workflow(example_cube_1,
  type = "obs_richness",
  dim_type = "map",
  level = "country",
  region = "Denmark"
)
diversity_map
```

`convert_eqdgc_latlong` *Convert extended quarter-degree grid cell codes to latitude and longitude*

Description

This is a helper function that takes a vector of EQDGC (Extended Quarter-Degree Grid Cell) codes and converts them to their central latitude and longitude coordinates. The function parses the base coordinates and the nested sub-grid codes to precisely locate the center of each grid cell.

Usage

```
convert_eqdgc_latlong(cellCode)
```

Arguments

`cellCode`	A character vector of EQDGC cell codes giving the longitude and latitude of a 1-degree cell (e.g. "E009N57" or "W010S34"), optionally followed by up to six letters A-D for sub-cells (e.g. "E009N57DD" = 0.25 degree cell).

Details

The function works by first extracting the base longitude and latitude coordinates from the cell code, accounting for direction (East/West and North/South). It then iteratively processes any sub-grid codes (e.g., `A`, `B`, `C`, `D`) to refine the coordinates. The final coordinates are the center point of the most specific grid cell.

Value

A matrix with two columns, `lat` and `long`, representing the central coordinates of each input grid cell.

Examples

```
# A 1-degree cell
convert_eqdgc_latlong("E010N10")

# A sub-grid cell
convert_eqdgc_latlong("W010S34ABCD")

# Multiple cell codes
convert_eqdgc_latlong(c("E010N10", "W010S34DDBA"))
```

`create_sf_from_utm` *Create an sf object from UTM coordinates, handling multiple zones correctly.*

Description

This function takes a data frame with UTM coordinates (xcoord, ycoord) and a utmzone column, and creates an sf object with the correct CRS for each zone.

Usage

```
create_sf_from_utm(df, output_crs = NULL)
```

Arguments

df	A data frame with numeric columns xcoord (easting), ycoord (northing) and utmzone, and a character column hemisphere ("S" for the southern hemisphere; any other value is treated as northern).
output_crs	(Optional) The EPSG code or CRS string for the desired output CRS. If NULL, the CRS of the first UTM zone will be used.

Value

A single sf object, with the points of each zone transformed to output_crs.

Examples

```
df <- data.frame(
  xcoord = c(500000, 501000),
  ycoord = c(5600000, 5601000),
  utmzone = c(32, 32),
  hemisphere = c("North", "North")
)
sf_obj <- create_sf_from_utm(df, output_crs = "EPSG:4326")
```

---

cum_richness_ts *Calculate Cumulative Species Richness*

Description

This function calculates cumulative species richness as a time series (see 'Details' for more information).

Usage

```
cum_richness_ts(data, ...)
```

Arguments

data	A data cube object (class 'processed_cube').
...	Arguments passed on to compute_indicator_workflow

cell_size (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- "grid" (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.

- "auto": determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as "10km" or "500m".

A manually selected cell size must be a whole number multiple of the cube's resolution.

level (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

region (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

ne_type (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

ne_scale (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

output_crs (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

first_year (Optional) Exclude data before this year. (Uses all data in the cube by default.)

last_year (Optional) Exclude data after this year. (Uses all data in the cube by default.)

spherical_geometry (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

make_valid (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

shapefile_path (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

shapefile_crs (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

invert (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

include_land (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Species richness:

Species richness is the total number of species present in a sample (Magurran, 1988). It is a fundamental and commonly used measure of biodiversity, providing a simple and intuitive overview of the status of biodiversity. However, richness is not well suited to measuring biodiversity change over time, as it only decreases when local extinctions occur and thus lags behind abundance for negative trends. While it may act as a leading indicator of alien species invasions, it will not indicate establishment because it ignores abundance. Nor will it necessarily indicate changes in local species composition, which can occur without any change in richness. Although richness is conceptually simple, it can be measured in different ways.

Cumulative richness:

Cumulative richness is calculated by adding the newly observed unique species each year to a cumulative sum. This indicator provides an estimation of whether and how many new species are still being discovered in a region. While an influx of alien species could cause an increase in cumulative richness, a fast-rising trend is likely an indication that the dataset is not comprehensive and therefore observed richness will provide an underestimate of species richness.

Value

An S3 object with the classes 'indicator_ts' and 'cum_richness' containing the calculated indicator values and metadata.

References

Magurran, A. E. (1988). *Ecological Diversity and Its Measurement*. Princeton University Press.

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
cr_ts <- cum_richness_ts(example_cube_1, first_year = 1985)
plot(cr_ts)
```

`example_cube_1` *Cube of GBIF Mammal Occurrences in Denmark*

Description

Small example cube containing the mammal occurrences in Denmark available on GBIF as of 16.03.2024.

Usage

```
example_cube_1
```

Format

A 'processed_cube' object: a list of metadata plus, in its `data` element, a tibble with 31,632 rows and 15 variables

year year occurrence was recorded

cellCode code containing the cell resolution and coordinates on the Extended Quarter Degree (eqdgc) grid

kingdomKey kingdom key associated with the species on GBIF

kingdom kingdom name

familyKey family key associated with the species on GBIF

family family name

taxonKey taxonomic key associated with the species on GBIF

scientificName scientific species name

obs number of occurrences (records)

minCoordinateUncertaintyInMeters minimum coordinate uncertainty in meters

minTemporalUncertainty minimum temporal uncertainty in seconds

familyCount number of occurrences of the family the species belongs to

xcoord longitude of the cell centre (degrees)

ycoord latitude of the cell centre (degrees)

resolution grid cell size (e.g. "0.25degrees")

Source

doi:10.15468/dl.5mb887

Examples

```
denmark_mammals_or_map <- obs_richness_map(example_cube_1,
                                           level = "country",
                                           region = "Denmark")
plot(denmark_mammals_or_map,
     title = "Mammals in Denmark (1862-2024): Observed Species Richness")
```

`example_indicator_map1` *Map of Observed Species Richness for Mammals in Denmark*

Description

Example indicator containing a map of observed species richness for mammal occurrences in Denmark (occurrences from GBIF: 16.03.2024).

Usage

```
example_indicator_map1
```

Format

An 'indicator_map' object: a list of metadata plus, in its `data` element, an sf data frame with 800 rows and 5 variables

cellid id of a map cell the indicator was calculated for

area area of the map cell in square kilometers

cellCode code containing the cell coordinates in Extended Quarter Degree (eqdgc) grid format

diversity_val calculated richness value for the cell

geometry geometry of the map cell

Source

doi:10.15468/dl.5mb887

example_indicator_ts1 *Time Series of Observed Species Richness for Mammals in Denmark*

Description

Example indicator containing a time series of observed species richness for mammal occurrences in Denmark, 1970-2024 (occurrences from GBIF: 16.03.2024).

Usage

```
example_indicator_ts1
```

Format

An 'indicator_ts' object: a list of metadata plus, in its `data` element, a tibble with 55 rows and 2 variables

year a year the indicator was calculated for

diversity_val calculated richness value for the year

Source

doi:10.15468/dl.5mb887

example_indicator_ts2 *Time Series of Cumulative Species Richness for Bryophytes in South Africa*

Description

Example indicator containing a time series of cumulative species richness for bryophyte occurrences in South Africa, 1875-2024 (occurrences from GBIF: 16.03.2024).

Usage

```
example_indicator_ts2
```

Format

An 'indicator_ts' object: a list of metadata plus, in its `data` element, a tibble with 110 rows and 2 variables

year a year the indicator was calculated for

diversity_val cumulative species richness up to and including the year

Source

doi:10.15468/dl.yfzgja

get_observed_years *Extract Years With Observations*

Description

Takes an "indicator_map", "indicator_ts", or "processed_cube" object and determines the years for which observation data exists.

Usage

```
get_observed_years(x)
```

Arguments

x	An "indicator_map", "indicator_ts" or "processed_cube" object.

Value

A data frame with two columns:

- `years`: A sequence of years covering the range of observations.
- `occurrences`: A logical vector indicating if observations exist for each year (`TRUE` if present, `FALSE` if absent).

Examples

```
total_occ_mammals_denmark <- total_occ_map(example_cube_1, level = "country",
                                           region = "Denmark")
get_observed_years(total_occ_mammals_denmark)
```

hill0_map *Calculate Estimated Hill Diversity Over Space or Time*

Description

Use coverage-based methods to estimate Hill diversity measures over a gridded map or as a time series. Three Hill diversity measures are covered:

Species richness - `hill0_map()` and `hill0_ts()`

Hill-Shannon diversity - `hill1_map()` and `hill1_ts()`

Hill-Simpson diversity - `hill2_map()` and `hill2_ts()`

(see 'Details' for more information).

Usage

```
hill0_map(
  data,
  coverage = 0.95,
  cutoff_length = 5,
  data_type = c("incidence", "abundance"),
  assume_freq = FALSE,
  ...
)

hill0_ts(data, coverage = 0.95, cutoff_length = 5, conf_level = 0.95, ...)

hill1_map(
  data,
  cutoff_length = 5,
  coverage = 0.95,
  data_type = c("incidence", "abundance"),
  assume_freq = FALSE,
  ...
)

hill1_ts(data, cutoff_length = 5, coverage = 0.95, conf_level = 0.95, ...)

hill2_map(
  data,
  cutoff_length = 5,
  coverage = 0.95,
  data_type = c("incidence", "abundance"),
  assume_freq = FALSE,
  ...
)

hill2_ts(data, cutoff_length = 5, coverage = 0.95, conf_level = 0.95, ...)
```

Arguments

`data`	A data cube object (class 'processed_cube').
`coverage`	(Optional) The sample coverage value for the estimator. Default is 0.95.
`cutoff_length`	(Optional) Minimum amount of data required, to avoid errors in the estimation. For maps, grid cells with fewer than `cutoff_length` species are removed. For time series, years whose species-by-cell incidence matrix has `cutoff_length` entries or fewer are removed. Default is 5.
`data_type`	(Optional, maps only) If set to "incidence", occurrences are converted to incidence data. Observations are treated as presence/absence and years are used as sampling units. The number of years in which a species was observed within a grid cell are then summarized. If set to "abundance", occurrences are summed across years for each species as a proxy for abundance. Default is "incidence".

`assume_freq` (Optional, maps only) If TRUE, the sum of observations for a species within a grid cell is assumed to be a sum of sampling sites in which that species was observed. The maximum number of observations for any species within a grid cell is then taken as a proxy for the total number of sampling sites for that cell. This parameter is ignored if data_type is set to "abundance". It is advisable to leave this on FALSE unless you are certain of how your data is structured. Default is FALSE.

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

`conf_level` (Optional, time series only) Confidence level for bootstrap confidence intervals. Only used when `num_bootstrap` > 0 and `ci_type` is not "none" (passed via `...`). Default is 0.95.

Details

Hill diversity:

Hill (1973) introduced the concept of Hill diversity, which assumes that the number and relative abundance of species are inseparable components of diversity. Hill diversity uses a single equation to calculate multiple measures of diversity by varying a single parameter, the order q, which changes the emphasis on rare vs common species (Roswell et al., 2021). It can be interpreted as the mean rarity of sampled species (the rarity of species i being 1/pi), and is calculated as:

$$D = \left(\sum_{i=1}^{S} p_i^q \right)^{1/(1-q)}$$

where D is diversity, S is the number of species, pi is the proportion of individuals belonging to species i, and q determines the rarity scale for the mean. While q can theoretically take almost any value, three common measures of diversity are special cases: species richness, and modified versions of the Shannon and Simpson diversity indices (Roswell et al., 2021). These three measures occur when q takes the value of 0, 1 (as the limit $q \to 1$, since the formula is undefined at exactly q = 1), or 2, respectively. (Roswell et al. (2021) use $\ell = 1 - q$.)

- **Species Richness (q = 0, `hill0`):**

$$D = S$$

- **Hill-Shannon Diversity (q = 1, `hill1`):**

$$D = e^{-\sum_{i=1}^{S} p_i \ln(p_i)}$$

- **Hill-Simpson Diversity (q = 2, `hill2`):**

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

Richness uses an arithmetic scale (the arithmetic mean), thus giving rare species a lot of leverage. By contrast, Hill-Shannon diversity uses a logarithmic scale (the geometric mean), treating common and rare species equally, and Hill-Simpson diversity uses a reciprocal scale (the harmonic mean), giving common species higher leverage.

Coverage-based estimation:

Hill diversity values can be estimated through different standardisation procedures as a way to mitigate the effects of sample size and sampling biases. One way to do this is by equalising sample size by calculating a species accumulation curve (a plot of cumulative species richness as a function of sample size) for each year or grid cell. The smallest sample size from among all the grid cells or years in the dataset is used as a reference to select richness values from each curve. This is called rarefaction. It is also possible to use a larger sample size as a reference, but this requires extrapolation of smaller samples, which is more prone to error than rarefaction.

However, results from sample-size based estimation can be problematic as they depend on both richness and evenness. A sample from a community with a more even distribution of individuals across species is likely to show higher richness than a sample of the same size from a community where many species are rare, as the rare species are less likely to appear in the sample. Similarly, a community containing a lot of species will appear less rich than it actually is if the sample size used for comparison is too small. Detectability also plays an important part; hard to detect species are less likely to appear in the sample, so communities in which rare species are more easily detectable are likely to yield richer samples.

Another way to estimate species richness is to standardise by coverage. The iNEXT package (Chao et al., 2014; Hsieh et al., 2016) for R is used to estimate species richness at an equal level of coverage (e.g. 0.95) for each cell or year in a biodiversity data cube. Coverage is the proportion of individuals in the community belonging to species in the sample. So, at a coverage of 0.95, 95% of individuals in the community belong to species detected in the sample while 5% belong to species that are not detected in the sample. Coverage is estimated based on the frequencies of species already in the sample. It can be illustrated using a species accumulation curve, the slope of which represents the probability of detecting a new species with the next individual you sample from a community. At a sample size of zero, the slope would be one, meaning the next individual sampled has a 100% probability of being a species not already in the sample. Therefore, a coverage value of one corresponds to the asymptote of a species accumulation curve (slope of zero), meaning no new species would be uncovered through further sampling.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'hill0' or 'hill1' or 'hill2' containing the calculated indicator values and metadata.

Functions

- `hill0_map()`:
- `hill0_ts()`:
- `hill1_map()`:
- `hill1_ts()`:
- `hill2_map()`:
- `hill2_ts()`:

References

Hill, M. O. (1973). Diversity and evenness: a unifying notation and its consequences. *Ecology*, *54*(2), 427-432.

Roswell, M., Dushoff, J., & Winfree, R. (2021). A conceptual guide to measuring species diversity. *Oikos*, *130*(3), 321-338.

Chao, A., Gotelli, N. J., Hsieh, T. C., Sander, E. L., Ma, K. H., Colwell, R. K., & Ellison, A. M. (2014). Rarefaction and extrapolation with Hill numbers: a framework for sampling and estimation in species diversity studies. *Ecological monographs*, *84*(1), 45-67.

Hsieh, T. C., Ma, K. H., & Chao, A. (2016). iNEXT: an R package for rarefaction and extrapolation of species diversity (Hill numbers). *Methods in Ecology and Evolution*, *7*(12), 1451-1456.

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
h0_map <- hill0_map(example_cube_1, level = "country", region = "Denmark")
plot(h0_map)


h0_ts <- hill0_ts(example_cube_1, first_year = 1985)
plot(h0_ts)


h1_map <- hill1_map(example_cube_1, level = "country", region = "Denmark")
plot(h1_map)


h1_ts <- hill1_ts(example_cube_1, first_year = 1985)
plot(h1_ts)


h2_map <- hill2_map(example_cube_1, level = "country", region = "Denmark")
```

```
plot(h2_map)


h2_ts <- hill2_ts(example_cube_1, first_year = 1985)
plot(h2_ts)
```

`list_species` *Extract Species Names.*

Description

Retrieves a standardized list of species names from a biodiversity data object ('processed_cube', 'indicator_map' or 'indicator_ts').

Usage

```
list_species(object)
```

Arguments

`object`	A biodiversity data object containing species names (either in a `data` slot or a `species_names` vector).

Value

A tibble with a single column:

- `scientificName`: The unique scientific names found in the object.

Examples

```
list_species(example_cube_1)
```

`mgrs_to_latlong_bbox` *Create a Longitude/Latitude Bounding Box from MGRS Data*

Description

Converts UTM coordinates of MGRS cells, which may span several UTM zones, to WGS 84 and returns their combined bounding box.

Usage

```
mgrs_to_latlong_bbox(df)
```

Arguments

df — A data frame with at least three columns: `cellCode` (MGRS codes; the first three characters give the UTM zone and latitude band), and `xcoord`/`ycoord` (UTM easting and northing).

Value

An `sf::st_bbox()` object in EPSG:4326 (longitude/latitude).

Examples

```
df <- data.frame(
  cellCode = c("32UUC", "32UUD"),
  xcoord = c(500000, 501000),
  ycoord = c(5600000, 5601000)
)
bbox <- mgrs_to_latlong_bbox(df)
```

---

newness_map — *Calculate Mean Year of Occurrence Over Time or Space*

Description

This function estimates the relative newness of records in a data cube by calculating the mean year of occurrence over a gridded map or as a time series (see 'Details' for more information).

Usage

```
newness_map(data, ...)

newness_ts(data, ...)
```

Arguments

data — A data cube object (class 'processed_cube').

... — Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.

- "auto": determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as "10km" or "500m".

A manually selected cell size must be a whole number multiple of the cube's resolution.

level (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

region (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

ne_type (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

ne_scale (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

output_crs (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

first_year (Optional) Exclude data before this year. (Uses all data in the cube by default.)

last_year (Optional) Exclude data after this year. (Uses all data in the cube by default.)

spherical_geometry (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

make_valid (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

shapefile_path (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

shapefile_crs (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

invert (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

include_land (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

The mean year of occurrence is calculated per cell, giving an indication of how recent the data is for each cell. A recent mean year is not necessarily an indication of quality, as some countries or regions have been conducting comprehensive biodiversity monitoring for many years and will therefore reflect an older mean year of occurrence, while others may show a recent mean year due to e.g., the sudden availability of large amounts of citizen science data.

For time series, the value for each year is the cumulative mean year of all records up to and including that year. Means are taken over cube records (one per taxon, cell and year), not weighted by occurrence counts. Values are rounded to the nearest year.

For maps, the optional argument `newness_min_year` (passed via `...`; default NULL) sets cells whose mean year is not above this value (e.g. 1970) to NA. This can be useful if outlier cells with very old data stretch the legend gradient so that other cell values are difficult to discern.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'newness' containing the calculated indicator values and metadata.

Functions

- `newness_map()`:
- `newness_ts()`:

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
n_map <- newness_map(example_cube_1, level = "country", region = "Denmark")
plot(n_map)


n_ts <- newness_ts(example_cube_1, first_year = 1985)
plot(n_ts)
```

`obs_richness_map` *Calculate Observed Species Richness Over Space or Time*

Description

This function calculates observed species richness over a gridded map or as a time series (see 'Details' for more information).

Usage

```
obs_richness_map(data, ...)

obs_richness_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Species richness:

Species richness is the total number of species present in a sample (Magurran, 1988). It is a fundamental and commonly used measure of biodiversity, providing a simple and intuitive overview of the status of biodiversity. However, richness is not well suited to measuring biodiversity change over time, as it only decreases when local extinctions occur and thus lags behind abundance for negative trends. While it may act as a leading indicator of alien species invasions, it will not indicate establishment because it ignores abundance. Nor will it necessarily indicate changes in local species composition, which can occur without any change in richness. Although richness is conceptually simple, it can be measured in different ways.

Observed richness:

Observed richness is calculated by counting the number of unique species observed for each year or each cell. Observed richness is highly dependent on the comprehensiveness of the dataset it is being applied to. If some regions are more intensively, carefully or systematically sampled than others, this will likely reflect as higher observed richness. Observed richness also depends on the relative abundance and spatial aggregation of each species, with less abundant and less aggregated species less likely to be discovered during surveys (Hillebrand et al., 2018), as well as the detectability of each species.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'obs_richness' containing the calculated indicator values and metadata.

Functions

- `obs_richness_map()`:
- `obs_richness_ts()`:

References

Magurran, A. E. (1988). *Ecological Diversity and Its Measurement*. Princeton University Press.

Hillebrand, H., Blasius, B., Borer, E. T., Chase, J. M., Downing, J. A., Eriksson, B. K., Filstrup, C. T., Harpole, W. S., Hodapp, D., Larsen, S., Lewandowska, A. M., Seabloom, E. W., Van de Waal, D. B., & Ryabov, A. B. (2018). Biodiversity change is uncoupled from species richness trends: Consequences for conservation and monitoring. *Journal of Applied Ecology*, *55*(1), 169-184.

See Also

compute_indicator_workflow(), add_ci()

Examples

```
or_map <- obs_richness_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(or_map)


or_ts <- obs_richness_ts(example_cube_1, first_year = 1985)
plot(or_ts)
```

---

occ_density_map — *Calculate Occurrence Density Over Space or Time*

Description

This function calculates the density of records over a gridded map or as a time series (see 'Details' for more information).

Usage

```
occ_density_map(data, ...)

occ_density_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to compute_indicator_workflow

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.

- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will

be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Density is the number of occurrences divided by area (km^2): the area of each grid cell for maps, and the total study-region area for time series. This provides similar information to total occurrences, but is adjusted for area.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'occ_density' containing the calculated indicator values and metadata.

Functions

- `occ_density_map()`:
- `occ_density_ts()`:

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
od_map <- occ_density_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(od_map)


od_ts <- occ_density_ts(example_cube_1, first_year = 1985)
plot(od_ts)
```

`occ_turnover_ts` *Calculate Occupancy Turnover*

Description

This function calculates occupancy turnover as a time series (see 'Details' for more information).

Usage

```
occ_turnover_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Occupancy turnover measures the change in species composition over time, reflecting the rate at which species appear or disappear from a given area. It provides insights into the dynamic nature

of ecological communities, highlighting shifts in species distributions and potential environmental changes. High turnover rates may indicate rapid community restructuring, potentially driven by factors such as habitat alteration, climate change, or invasive species. Analyzing occupancy turnover can be crucial for understanding ecosystem stability, identifying areas of conservation concern, and assessing the effectiveness of management strategies.

Occupancy turnover can be calculated in different ways, but here we use the Jaccard dissimilarity index (Jaccard, 1901) to measure the dissimilarity between the species sets of consecutive years (the first year is NA). The Jaccard dissimilarity is calculated as:

$$J = (b + c)/(a + b + c)$$

where a is the number of species present in both time periods, b is the number of species present only in the first time period, and c is the number of species present only in the second time period. The index ranges from 0 (no turnover) to 1 (complete turnover).

Value

An S3 object with the classes 'indicator_ts' and 'occ_turnover' containing the calculated indicator values and metadata.

References

Jaccard, P. (1901). Étude de la distribution florale dans une portion des Alpes et du Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, *37*(142), 547-579.

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
ot_ts <- occ_turnover_ts(example_cube_1, first_year = 1985)
plot(ot_ts)
```

`pielou_evenness_map` *Calculate Evenness Over Time or Space*

Description

Calculate evenness over a gridded map or as a time series (see 'Details' for more information).

Usage

```
pielou_evenness_map(data, ...)

pielou_evenness_ts(data, ...)

williams_evenness_map(data, ...)

williams_evenness_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Evenness:

Species evenness is a commonly used indicator that measures how uniformly individuals are distributed across species in a region or over time. It provides a complement to richness by taking relative abundance into account. Although GBIF provides information about abundances as individual counts, the majority of entries lack this information. Hence, evenness can only be calculated using the proportions of observations rather than proportions of individuals. Strictly

speaking, the evenness measures therefore indicate how uniformly species are represented in the respective data set rather than the true evenness of the ecological community.

Pielou's evenness:

Pielou's evenness (1966) is a well-known and commonly used evenness measure. It is calculated as:

$$E = \frac{-\sum_{i=1}^{S} p_i \ln(p_i)}{\ln(S)}$$

where S is the number of species observed in the grid cell (maps) or year (time series) and pi is the proportion of occurrences represented by species i. Species absent from a cell or year do not count towards S, and evenness is NA when fewer than two species are present.

Williams' evenness:

An analysis of evenness properties by Kvålseth (2015) showed that an evenness index introduced by Williams in 1977 in an unpublished manuscript has two important properties which Pielou's does not. The properties in question are complex mathematical properties known as the Schur-Concavity and value validity, but we attempt to describe them here more simply. If a measure of evenness is Schur-concave, it means that when the distribution of individuals becomes more evenly spread across species, the measure of evenness will stay the same or increase, but never decrease. Value validity means that an evenness index should provide sensible and meaningful values across its range for any given distribution of species abundances. Kvålseth referred to this evenness measure as E9 but we refer to it as Williams' evenness.

Williams' evenness is calculated as:

$$1 - \sqrt{\frac{S \sum_{i=1}^{S} p_i^2 - 1}{S - 1}}$$

where S is the number of species observed in the grid cell (maps) or year (time series) and pi is the proportion of occurrences represented by species i. Species absent from a cell or year do not count towards S, and evenness is NA when fewer than two species are present.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'pielou_evenness' or 'williams_evenness' containing the calculated indicator values and metadata.

Functions

- `pielou_evenness_map()`:
- `pielou_evenness_ts()`:
- `williams_evenness_map()`:
- `williams_evenness_ts()`:

References

Pielou, E. C. (1966). The measurement of diversity in different types of biological collections. *Journal of theoretical biology*, *13*, 131-144.

Kvålseth, T. O. (2015). Evenness indices once again: critical analysis of properties. *SpringerPlus*, *4*, 1-12.

See Also

compute_indicator_workflow(), add_ci()

Examples

```
pe_map <- pielou_evenness_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(pe_map)


pe_ts <- pielou_evenness_ts(example_cube_1, first_year = 1985)
plot(pe_ts)


we_map <- williams_evenness_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(we_map)


we_ts <- williams_evenness_ts(example_cube_1, first_year = 1985)
plot(we_ts)
```

---

plot_map *Plot Biodiversity Indicator Map*

Description

Creates a map visualization of a calculated multi-species biodiversity indicator. Requires an indicator_map object as input. To plot single-species indicators, use the plot_species_map() function instead.

Usage

```
plot_map(
  x,
  title = "auto",
```

```
  auto_title = NULL,
  leg_label_default = NULL,
  xlims = NULL,
  ylims = NULL,
  trans = NULL,
  bcpower = NULL,
  breaks = NULL,
  labels = NULL,
  output_crs = NULL,
  crop_to_grid = NULL,
  crop_by_region = FALSE,
  ocean_fill_colour = NULL,
  land_fill_colour = NULL,
  grid_fill_colour = NULL,
  grid_line_colour = NULL,
  grid_line_width = NULL,
  grid_fill_transparency = NULL,
  grid_line_transparency = NULL,
  legend_title = NULL,
  legend_limits = NULL,
  legend_title_wrap_length = 10,
  title_wrap_length = 60,
  visible_gridlines = TRUE,
  visible_panel_gridlines = FALSE,
  map_expansion_factor = 0.1,
  layers = NULL,
  layer_colours = NULL,
  layer_fill_colours = NULL,
  scale = c("medium", "small", "large"),
  filter_outliers = FALSE
)
```

Arguments

`x` An 'indicator_map' object containing multi-species indicator values associated with map grid cells. This is a required parameter with no default.

`title` (Optional) Plot title. Replace "auto" with your own title if you want a custom title or if calling the function manually.

`auto_title` (Optional) Text for automatic title generation, provided by an appropriate S3 method (if calling the function manually, leave as NULL).

`leg_label_default` (Optional) Default label for the legend, provided by an appropriate S3 method (if calling the function manually, leave as NULL).

`xlims` (Optional) Custom longitude limits in decimal degrees (WGS84), as c(min, max). Should be supplied together with ylims. If only xlims is supplied, the values are instead interpreted in the units of the indicator_map's coordinate reference system and combined with the map's own y limits.

`ylims` (Optional) Custom latitude limits in decimal degrees (WGS84), as c(min, max). Should be supplied together with xlims. If only ylims is supplied, the values are instead interpreted in the units of the indicator_map's coordinate reference system and combined with the map's own x limits.

`trans` (Optional) Scale transformation for the fill gradient. Can be any transformation accepted by `ggplot2::scale_fill_gradient()` (e.g., 'log', 'log10' or 'sqrt'), or one of the special values 'boxcox', 'modulus' or 'yj' (Yeo-Johnson), which use the power parameter given in bcpower.

`bcpower` (Optional) Power parameter for the Box-Cox, modulus, or Yeo-Johnson transformations (used only when trans is 'boxcox', 'modulus' or 'yj').

`breaks` (Optional) Break points for the legend scale.

`labels` (Optional) Labels for legend scale break points.

`output_crs` (Optional) Coordinate Reference System (CRS) for the output map. Can be specified as an EPSG code (e.g., 4326) or a proj4string. If NULL (default), the original CRS of the indicator_map object will be used.

`crop_to_grid` (Optional) If TRUE, the grid will determine the edges of the map. If FALSE, a buffer will be added around the grid. If NULL (default), will be set to TRUE if map_level is "cube", otherwise FALSE.

`crop_by_region` (Optional) If TRUE, the map extent is set to the bounding box of the region that was specified when calculating the indicator_map (e.g. the country or continent), instead of the extent of the grid. This requires that a region was specified, i.e. that the indicator_map was not calculated with level = "cube" or level = "world". Default is FALSE.

`ocean_fill_colour` (Optional) Colour for the ocean (plot background) outside of the grid. Default is "#92c5f0" (light blue).

`land_fill_colour` (Optional) Colour for the land area outside of the grid. Default is "grey85".

`grid_fill_colour` (Optional) Colour for empty grid cells (non-empty grid cells will be coloured according to their indicator value). Default is "transparent".

`grid_line_colour` (Optional) Colour for the grid lines. Default is "black". If visible_gridlines is set to FALSE, this setting will have no effect.

`grid_line_width` (Optional) Width of the grid lines. If NULL (default), 0.5 for ISEA3H grids and 0.1 otherwise.

`grid_fill_transparency` (Optional) Transparency of the grid fill colour for empty grid cells (0 = fully transparent, 1 = fully opaque). If visible_gridlines is set to TRUE, default is 0.2. Otherwise, default is 0. Note that this setting does NOT apply to grid cells with indicator values, and has no visible effect while grid_fill_colour is "transparent".

`grid_line_transparency` (Optional) Transparency of the grid line colour (0 = fully transparent, 1 = fully opaque). Default is 0.5. If visible_gridlines is set to FALSE, this setting will have no effect. *Note that this setting does NOT apply to the grid outline!

`legend_title` (Optional) Title for the plot legend.

`legend_limits` (Optional) Limits for the legend scale.

`legend_title_wrap_length`
(Optional) Maximum legend title length before wrapping to a new line.

`title_wrap_length`
(Optional) Maximum title length before wrapping to a new line.

`visible_gridlines`
(Optional) Show gridlines between cells. Default is TRUE.

`visible_panel_gridlines`
(Optional) Show ggplot panel gridlines. Default is FALSE.

`map_expansion_factor`
(Optional) Factor to expand the map limits beyond the grid limits. This does NOT expand the boundaries of the plot, it only affects where the crop is applied. If this value is too small, some land may be visibly cut off due to map distortion caused by projections. A larger value will extend the bounding box for cropping to prevent this. Must be a positive number. Default is 0.1. This should be enough for most projections, but you can increase this value if you are using an extreme projection and find that some land is visibly cut off.

`layers` (Optional) Additional rnaturalearth layers to plot, e.g. c("reefs", "playas").

`layer_colours` (Optional) Outline colours for the additional layers, given in the same order as 'layers' (one colour per layer; must be the same length as 'layers'). If NULL (default), all layer outlines are black.

`layer_fill_colours`
(Optional) Fill colours for the additional layers, given in the same order as 'layers' (must be the same length as 'layers'). If NULL (default), layers are unfilled, except "ocean" and "lakes", which are filled light blue.

`scale` (Optional) Scale of Natural Earth data ("small", "medium", or "large"). Default is 'medium'.

`filter_outliers`
(Optional) If TRUE, removes geographical outliers from the plot extent using an Interquartile Range (IQR) method based on the centroid coordinates of the indicator_map cells. This is particularly useful for discrete global grids like ISEA3H where stray data points from coordinate errors can force the map bounds to expand globally. Default is FALSE.

Value

A ggplot object representing the biodiversity indicator map. Can be customized using ggplot2 functions.

Examples

```
evenness_map <- pielou_evenness_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot_map(
```

```
  x = evenness_map,
  title = "Map of Species Evenness in Denmark",
  legend_title = "Evenness"
)
```

---

plot_mv *Plot Interactive Biodiversity Indicator Map*

Description

Creates an interactive map of a calculated multi-species biodiversity indicator. Requires an indicator_map object as input.

Usage

```
plot_mv(
  x,
  legend_title = NULL,
  transparency = 0.8,
  filter_NA = TRUE,
  basemap_list = c("OpenStreetMap", "OpenTopoMap", "CartoDB.Positron",
    "Esri.WorldImagery"),
  ...
)
```

Arguments

`x`	An 'indicator_map' object containing multi-species indicator values associated with map grid cells. This is a required parameter with no default.
`legend_title`	(Optional) Legend title. Add your own custom legend title if you don't like the automatically generated one. If set to the default of NULL, the function will provide a legend title based on the indicator being plotted.
`transparency`	(Optional) Change the transparency of the grid fill. Default is 0.8. A value of 0 is full transparency, 1 is solid fill.
`filter_NA`	(Optional) Filter out NA values so NA grid cells do not appear. Default is TRUE.
`basemap_list`	(Optional) A list of basemaps you want mapview to display as options to select from. The first one in the list will be shown when the map loads. Default is c("OpenStreetMap", "OpenTopoMap", "CartoDB.Positron", "Esri.WorldImagery").
`...`	(Optional) Additional arguments passed to `mapview::mapview()`.

Value

An interactive mapview plot.

Examples

```
# Opens an interactive map, so only run in interactive sessions
if (interactive() && requireNamespace("mapview", quietly = TRUE) &&
    requireNamespace("RColorBrewer", quietly = TRUE)) {
  cube_path <- system.file("extdata", "denmark_mammals_cube_eea.csv",
                           package = "b3gbi")
  cube <- process_cube(cube_path)
  map_rich <- obs_richness_map(cube, level = "country", region = "Denmark")
  plot_mv(map_rich)
}
```

`plot_species_map` *Plot Occurrence Map or Range Map of Individual Species*

Description

Creates map visualizations of species ranges or species occurrences. Requires an indicator_map object created using the `spec_occ_map()` or `spec_range_map()` functions as input. It is also the function called by `plot()` for species-level maps, including those created with `relative_occupancy_map()`. To plot multi-species indicators (e.g., species richness or evenness), use the `plot_map()` function instead.

Usage

```
plot_species_map(
  x,
  species,
  single_plot = TRUE,
  title = "auto",
  auto_title = NULL,
  leg_label_default = NULL,
  suppress_legend = FALSE,
  xlims = NULL,
  ylims = NULL,
  trans = NULL,
  bcpower = NULL,
  breaks = NULL,
  labels = NULL,
  output_crs = NULL,
  crop_to_grid = NULL,
  crop_by_region = FALSE,
  ocean_fill_colour = NULL,
  land_fill_colour = NULL,
  grid_fill_colour = NULL,
  grid_line_colour = NULL,
```

```
  grid_line_width = NULL,
  grid_fill_transparency = NULL,
  grid_line_transparency = NULL,
  legend_title = NULL,
  legend_limits = NULL,
  legend_title_wrap_length = 10,
  title_wrap_length = 60,
  spec_name_wrap_length = 40,
  visible_gridlines = TRUE,
  visible_panel_gridlines = FALSE,
  map_expansion_factor = 0.5,
  layers = NULL,
  layer_colours = NULL,
  layer_fill_colours = NULL,
  scale = c("medium", "small", "large"),
  filter_outliers = FALSE
)
```

Arguments

`x` An 'indicator_map' object containing indicator values for individual species associated with map grid cells. This object is typically created using the `spec_occ_map()`, `spec_range_map()` or `relative_occupancy_map()` functions. This is a required parameter with no default.

`species` Species you want to map. Can be either numerical taxonKeys or species names. Partial species names can be given as the beginning of a name (prefix match). This is a required parameter with no default.

`single_plot` (Optional) If TRUE, all species maps will be combined into a single multi-panel plot. Set this to FALSE to plot each species separately. Default is TRUE.

`title` (Optional) Plot title. Replace "auto" with your own title if you want a custom title or if calling the function manually.

`auto_title` (Optional) Text for automatic title generation, provided by an appropriate S3 method (if calling the function manually, leave as NULL).

`leg_label_default` (Optional) Default label for the legend, provided by an appropriate S3 method (if calling the function manually, leave as NULL).

`suppress_legend` (Optional) Do not show legend. This defaults to FALSE but will be forcibly set to TRUE when plotting species ranges, as all cell values are 1.

`xlims` (Optional) Custom longitude limits in decimal degrees (WGS84), as c(min, max). Should be supplied together with ylims. If only xlims is supplied, the values are instead interpreted in the units of the indicator_map's coordinate reference system and combined with the map's own y limits.

`ylims` (Optional) Custom latitude limits in decimal degrees (WGS84), as c(min, max). Should be supplied together with xlims. If only ylims is supplied, the values are instead interpreted in the units of the indicator_map's coordinate reference system and combined with the map's own x limits.

`trans` (Optional) Scale transformation for the fill gradient. Can be any transformation accepted by `ggplot2::scale_fill_gradient()` (e.g., 'log', 'log10' or 'sqrt'), or one of the special values 'boxcox', 'modulus' or 'yj' (Yeo-Johnson), which use the power parameter given in bcpower.

`bcpower` (Optional) Power parameter for the Box-Cox, modulus, or Yeo-Johnson transformations (used only when trans is 'boxcox', 'modulus' or 'yj').

`breaks` (Optional) Break points for the legend scale.

`labels` (Optional) Labels for legend scale break points.

`output_crs` (Optional) Coordinate Reference System (CRS) for the output map. Can be specified as an EPSG code (e.g., 4326) or a proj4string. If NULL (default), the original CRS of the indicator_map object will be used.

`crop_to_grid` (Optional) If TRUE, the grid will determine the edges of the map. If FALSE, a buffer will be added around the grid. If NULL (default), will be set to TRUE if map_level is "cube", otherwise FALSE.

`crop_by_region` (Optional) If TRUE, the map extent is set to the bounding box of the region that was specified when calculating the indicator_map (e.g. the country or continent), instead of the extent of the grid. This requires that a region was specified, i.e. that the indicator_map was not calculated with level = "cube" or level = "world". Default is FALSE.

`ocean_fill_colour` (Optional) Colour for the ocean (plot background) outside of the grid. Default is "#92c5f0" (light blue).

`land_fill_colour` (Optional) Colour for the land area outside of the grid. Default is "grey85".

`grid_fill_colour` (Optional) Colour for empty grid cells (non-empty grid cells will be coloured according to their indicator value). Default is "transparent".

`grid_line_colour` (Optional) Colour for the grid lines. Default is "black". If visible_gridlines is set to FALSE, this setting will have no effect.

`grid_line_width` (Optional) Width of the grid lines. If NULL (default), 0.5 for ISEA3H grids and 0.1 otherwise.

`grid_fill_transparency` (Optional) Transparency of the grid fill colour for empty grid cells (0 = fully transparent, 1 = fully opaque). If visible_gridlines is set to TRUE, default is 0.2. Otherwise, default is 0. Note that this setting does NOT apply to grid cells with indicator values, and has no visible effect while grid_fill_colour is "transparent".

`grid_line_transparency` (Optional) Transparency of the grid line colour (0 = fully transparent, 1 = fully opaque). Default is 0.5. If visible_gridlines is set to FALSE, this setting will have no effect. *Note that this setting does NOT apply to the grid outline!

`legend_title` (Optional) Title for the plot legend.

`legend_limits` (Optional) Limits for the legend scale.

`legend_title_wrap_length`
: (Optional) Maximum legend title length before wrapping to a new line.

`title_wrap_length`
: (Optional) Maximum title length before wrapping to a new line.

`spec_name_wrap_length`
: (Optional) Maximum species name length before wrapping to a new line. Default: 40 characters.

`visible_gridlines`
: (Optional) Show gridlines between cells. Default is TRUE.

`visible_panel_gridlines`
: (Optional) Show ggplot panel gridlines. Default is FALSE.

`map_expansion_factor`
: (Optional) Factor to expand the map limits beyond the grid limits. This does NOT expand the boundaries of the plot, it only affects where the crop is applied. If this value is too small, some land may be visibly cut off due to map distortion caused by projections. A larger value will extend the bounding box for cropping to prevent this. Must be a positive number. Default is 0.5. You can increase this value if you are using an extreme projection and find that some land is visibly cut off.

`layers`
: (Optional) Additional rnaturalearth layers to plot, e.g. c("reefs", "playas").

`layer_colours`
: (Optional) Outline colours for the additional layers, given in the same order as 'layers' (one colour per layer; must be the same length as 'layers'). If NULL (default), all layer outlines are black.

`layer_fill_colours`
: (Optional) Fill colours for the additional layers, given in the same order as 'layers' (must be the same length as 'layers'). If NULL (default), layers are unfilled, except "ocean" and "lakes", which are filled light blue.

`scale`
: (Optional) Scale of Natural Earth data ("small", "medium", or "large"). Default is 'medium'.

`filter_outliers`
: (Optional) If TRUE, removes geographical outliers from the plot extent using an Interquartile Range (IQR) method based on the centroid coordinates of the indicator_map cells. This is particularly useful for discrete global grids like ISEA3H where stray data points from coordinate errors can force the map bounds to expand globally. Default is FALSE.

Value

If single_plot = TRUE (default), a patchwork object combining one ggplot per species. If single_plot = FALSE, a named list of ggplot objects (one per species). These can be customized using ggplot2 and patchwork functions. Requires the 'patchwork' package.

Examples

```
if (requireNamespace("patchwork", quietly = TRUE)) {
  spec_occ_mammals_denmark <- spec_occ_map(example_cube_1,
    level = "country",
```

```
    region = "Denmark"
  )
  plot_species_map(x = spec_occ_mammals_denmark, c(2440728, 4265185))
}
```

---

plot_species_ts *Plot Occurrence Trends or Range Size Trends for Individual Species*

Description

Creates time series plots of species occurrences or species range sizes, with an optional smoothed trendline, and visualizes uncertainty. Requires an indicator_ts object created using the `spec_occ_ts()` or `spec_range_ts()` functions as input. It is also the function called by `plot()` for species-level time series, including those created with `relative_occupancy_ts()`. To plot multi-species indicators (e.g., species richness or evenness), use the `plot_ts()` function instead.

Usage

```
plot_species_ts(
  x,
  species,
  single_plot = TRUE,
  min_year = NULL,
  max_year = NULL,
  title = "auto",
  auto_title = NULL,
  y_label_default = NULL,
  suppress_y = FALSE,
  smoothed_trend = TRUE,
  linecolour = NULL,
  linealpha = 0.8,
  ribboncolour = NULL,
  ribbonalpha = 0.2,
  error_alpha = 1,
  trendlinecolour = NULL,
  trendlinealpha = 0.5,
  envelopecolour = NULL,
  envelopealpha = 0.2,
  smooth_cialpha = 1,
  point_line = c("point", "line"),
  pointsize = 2,
  linewidth = 1,
  ci_type = c("error_bars", "ribbon"),
  error_width = 1,
  error_thickness = 1,
```

```
  smooth_linetype = c("solid", "dashed", "dotted", "dotdash", "longdash", "twodash"),
   smooth_linewidth = 1,
   smooth_cilinewidth = 1,
   gridoff = FALSE,
   x_label = NULL,
   y_label = NULL,
   x_expand = 0.1,
   y_expand = 0.1,
   x_breaks = 10,
   y_breaks = 6,
   title_wrap_length = 60,
   spec_name_wrap_length = 40
)
```

Arguments

x	An 'indicator_ts' object containing time series of indicator values matched to species names and/or taxon keys, created using the `spec_occ_ts()`, `spec_range_ts()` or `relative_occupancy_ts()` functions. This is a required parameter with no default.
species	Species you want to plot. Can be either numerical taxonKeys or species names. Partial species names can be given as the beginning of a name (prefix match). This is a required parameter with no default.
single_plot	(Optional) If TRUE (default), all species time series will be combined into a single multi-panel plot. Set this to FALSE to plot each species separately.
min_year	(Optional) Earliest year to include in the plot.
max_year	(Optional) Latest year to include in the plot.
title	(Optional) Plot title. Replace "auto" with your own title if you want a custom title or if calling the function manually.
auto_title	(Optional) Text for automatic title generation, provided by an appropriate S3 method (if calling the function manually, leave as NULL).
y_label_default	(Optional) Default label for the y-axis, provided by an appropriate S3 method (if calling the function manually, leave as NULL).
suppress_y	(Optional) If TRUE, suppresses y-axis tick labels. Default is FALSE.
smoothed_trend	(Optional) If TRUE, plot a smoothed trendline over time (`stats::loess()`). Default is TRUE.
linecolour	(Optional) Colour for the indicator line or points. Default is darkorange.
linealpha	(Optional) Transparency for indicator line or points. Default is 0.8.
ribboncolour	(Optional) Colour for the bootstrapped confidence intervals. Default is goldenrod1. Set to "NA" if you don't want to plot the CIs.
ribbonalpha	(Optional) Transparency for indicator confidence interval ribbon (if ci_type = "ribbon"). Default is 0.2.
error_alpha	(Optional) Transparency for indicator error bars (if ci_type = "error_bars"). Default is 1.

`trendlinecolour`
: (Optional) Colour for the smoothed trendline. Default is blue.

`trendlinealpha`
: (Optional) Transparency for the smoothed trendline. Default is 0.5.

`envelopecolour`
: (Optional) Colour for the uncertainty envelope. Default is "lightsteelblue1".

`envelopealpha`
: (Optional) Transparency for the smoothed trendline envelope. Default is 0.2.

`smooth_cialpha`
: (Optional) Transparency for the smoothed lines forming the edges of the trendline envelope. Default is 1.

`point_line`
: (Optional) Whether to plot the indicator as a line or a series of points. Options are "line" or "point". Default is "point".

`pointsize`
: (Optional) Size of the points if point_line = "point". Default is 2.

`linewidth`
: (Optional) Width of the line if point_line = "line". Default is 1.

`ci_type`
: (Optional) Whether to plot bootstrapped confidence intervals as a "ribbon" or "error_bars". Default is "error_bars".

`error_width`
: (Optional) Width of error bars if ci_type = "error_bars". Default is 1. Note that unlike the default 'width' parameter in geom_errorbar, 'error_width' is NOT dependent on the number of data points in the plot. It is automatically scaled to account for this. Therefore the width you select will be consistent relative to the plot width even if you change 'min_year' and 'max_year'.

`error_thickness`
: (Optional) Thickness of error bars if ci_type = "error_bars". Default is 1.

`smooth_linetype`
: (Optional) Type of line to plot for smoothed trendline. Options are "solid", "dashed", "dotted", "dotdash", "longdash" or "twodash". Default is "solid".

`smooth_linewidth`
: (Optional) Line width for smoothed trendline. Default is 1.

`smooth_cilinewidth`
: (Optional) Line width for smoothed trendline confidence intervals. Default is 1.

`gridoff`
: (Optional) If TRUE, hides gridlines.

`x_label`
: (Optional) Label for the x-axis.

`y_label`
: (Optional) Label for the y-axis.

`x_expand`
: (Optional) Expansion factor to expand the x-axis beyond the data. Left and right values are required in the form of c(0.1, 0.2) or simply 0.1 to apply the same value to each side. Default is 0.1.

`y_expand`
: (Optional) Expansion factor to expand the y-axis beyond the data. Lower and upper values are required in the form of c(0.1, 0.2) or simply 0.1 to apply the same value to the top and bottom. Default is 0.1.

`x_breaks`
: (Optional) Integer giving desired number of breaks for x axis. (May not return exactly the number requested.) Default is 10.

`y_breaks`
: (Optional) Integer giving desired number of breaks for y axis. (May not return exactly the number requested.) Default is 6.

`title_wrap_length`
: (Optional) Maximum title length before wrapping to a new line.

`spec_name_wrap_length`
: (Optional) Maximum species name length before wrapping to a new line. Default is 40.

Value

If single_plot = TRUE (default), a patchwork object combining one ggplot per species. If single_plot = FALSE, a named list of ggplot objects (one per species); if only one species is plotted, a patchwork object is returned regardless. These can be customized using ggplot2 and patchwork functions. Requires the 'patchwork' package.

Examples

```
if (requireNamespace("patchwork", quietly = TRUE)) {
  spec_occ_ts_mammals_denmark <- spec_occ_ts(example_cube_1,
                                             level = "country",
                                             region = "Denmark")
  # default colours:
  plot_species_ts(spec_occ_ts_mammals_denmark, c(2440728, 4265185))

  # custom colours:
  plot_species_ts(spec_occ_ts_mammals_denmark, c(2440728, 4265185),
          linecolour = "thistle",
          trendlinecolour = "forestgreen",
          envelopecolour = "lightgreen")
}
```

---

plot_ts *Plot Biodiversity Indicator Trend*

Description

Creates a time series plot of a calculated multi-species biodiversity indicator (e.g. species richness or evenness), with an optional smoothed trendline, and visualizes uncertainty. Requires an 'indicator_ts' object as input. To plot a single-species indicator time series, use the `plot_species_ts()` function instead.

Usage

```
plot_ts(
  x,
  min_year = NULL,
  max_year = NULL,
  title = "auto",
  auto_title = NULL,
  y_label_default = NULL,
  suppress_y = FALSE,
  smoothed_trend = TRUE,
  linecolour = NULL,
  linealpha = 0.8,
  ribboncolour = NULL,
  ribbonalpha = 0.2,
```

```
  error_alpha = 1,
  trendlinecolour = NULL,
  trendlinealpha = 0.5,
  envelopecolour = NULL,
  envelopealpha = 0.2,
  smooth_cialpha = 1,
  point_line = c("point", "line"),
  pointsize = 2,
  linewidth = 1,
  ci_type = c("error_bars", "ribbon"),
  error_width = 1,
  error_thickness = 1,
 smooth_linetype = c("solid", "dashed", "dotted", "dotdash", "longdash", "twodash"),
  smooth_linewidth = 1,
  smooth_cilinewidth = 1,
  gridoff = FALSE,
  x_label = NULL,
  y_label = NULL,
  x_expand = 0.1,
  y_expand = 0.1,
  x_breaks = 10,
  y_breaks = 6,
  title_wrap_length = 60
)
```

Arguments

`x`	An 'indicator_ts' object containing a time series of multi-species indicator values. This is a required parameter with no default.
`min_year`	(Optional) Earliest year to include in the plot.
`max_year`	(Optional) Latest year to include in the plot.
`title`	(Optional) Plot title. Replace "auto" with your own title if you want a custom title or if calling the function manually.
`auto_title`	(Optional) Text for automatic title generation, provided by an appropriate S3 method (if calling the function manually, leave as NULL).
`y_label_default`	(Optional) Default label for the y-axis, provided by an appropriate S3 method (if calling the function manually, leave as NULL).
`suppress_y`	(Optional) If TRUE, suppresses y-axis tick labels. Default is FALSE.
`smoothed_trend`	(Optional) If TRUE, plot a smoothed trendline over time (`stats::loess()`). Default is TRUE.
`linecolour`	(Optional) Colour for the indicator line or points. Default is darkorange.
`linealpha`	(Optional) Transparency for indicator line or points. Default is 0.8.
`ribboncolour`	(Optional) Colour for the bootstrapped confidence intervals. Default is goldenrod1. Set to "NA" if you don't want to plot the CIs.

`ribbonalpha` (Optional) Transparency for indicator confidence interval ribbon (if ci_type = "ribbon"). Default is 0.2.

`error_alpha` (Optional) Transparency for indicator error bars (if ci_type = "error_bars"). Default is 1.

`trendlinecolour` (Optional) Colour for the smoothed trendline. Default is blue.

`trendlinealpha` (Optional) Transparency for the smoothed trendline. Default is 0.5.

`envelopecolour` (Optional) Colour for the uncertainty envelope. Default is "lightsteelblue1".

`envelopealpha` (Optional) Transparency for the smoothed trendline envelope. Default is 0.2.

`smooth_cialpha` (Optional) Transparency for the smoothed lines forming the edges of the trendline envelope. Default is 1.

`point_line` (Optional) Whether to plot the indicator as a line or a series of points. Options are "line" or "point". Default is "point".

`pointsize` (Optional) Size of the points if point_line = "point". Default is 2.

`linewidth` (Optional) Width of the line if point_line = "line". Default is 1.

`ci_type` (Optional) Whether to plot bootstrapped confidence intervals as a "ribbon" or "error_bars". Default is "error_bars".

`error_width` (Optional) Width of error bars if ci_type = "error_bars". Default is 1. Note that unlike the default 'width' parameter in geom_errorbar, 'error_width' is NOT dependent on the number of data points in the plot. It is automatically scaled to account for this. Therefore the width you select will be consistent relative to the plot width even if you change 'min_year' and 'max_year'.

`error_thickness` (Optional) Thickness of error bars if ci_type = "error_bars". Default is 1.

`smooth_linetype` (Optional) Type of line to plot for smoothed trendline. Options are "solid", "dashed", "dotted", "dotdash", "longdash" or "twodash". Default is "solid".

`smooth_linewidth` (Optional) Line width for smoothed trendline. Default is 1.

`smooth_cilinewidth` (Optional) Line width for smoothed trendline confidence intervals. Default is 1.

`gridoff` (Optional) If TRUE, hides gridlines.

`x_label` (Optional) Label for the x-axis.

`y_label` (Optional) Label for the y-axis.

`x_expand` (Optional) Expansion factor to expand the x-axis beyond the data. Left and right values are required in the form of c(0.1, 0.2) or simply 0.1 to apply the same value to each side. Default is 0.1.

`y_expand` (Optional) Expansion factor to expand the y-axis beyond the data. Lower and upper values are required in the form of c(0.1, 0.2) or simply 0.1 to apply the same value to the top and bottom. Default is 0.1.

`x_breaks` (Optional) Integer giving desired number of breaks for x axis. (May not return exactly the number requested.) Default is 10.

`y_breaks` (Optional) Integer giving desired number of breaks for y axis. (May not return exactly the number requested.) Default is 6.

`title_wrap_length` (Optional) Maximum title length before wrapping to a new line.

Value

A ggplot object representing the biodiversity indicator time series plot. Can be customized using ggplot2 functions.

Examples

```
# default colours:
plot_ts(example_indicator_ts1,
        y_label = "Species Richness",
        title = "Observed Species Richness: Mammals in Denmark")

# custom colours:
plot_ts(example_indicator_ts1,
        y_label = "Species Richness",
        title = "Observed Species Richness: Mammals in Denmark",
        linecolour = "thistle",
        trendlinecolour = "forestgreen",
        envelopecolour = "lightgreen")
```

---

prepare_indicator_bootstrap

Prepare bootstrap and confidence interval parameters for an indicator

---

Description

This function prepares the argument lists for `dubicube::bootstrap_cube()` and `dubicube::calculate_bootstrap_ci()` based on the indicator definition. Behaviour (grouping, bootstrap method, transformations, bias correction) is fully controlled by a rule book keyed on `indicator$div_type`.

Usage

```
prepare_indicator_bootstrap(
  indicator,
  num_bootstrap,
  ci_type,
  seed = 123,
  trans = function(t) t,
  inv_trans = function(t) t,
  confidence_level = 0.95,
  expected_years = NULL,
  boot_args = list(),
  ci_args = list()
)
```

Arguments

`indicator`	A list describing the indicator. Must contain at least `div_type` and `raw_data`.
`num_bootstrap`	Integer. Number of bootstrap samples.
`ci_type`	Character. Type of confidence interval.
`seed`	Optional integer. Random seed for bootstrapping. (Default: 123)
`trans`	Optional transformation function applied to the statistic. Will be overridden if specified by the indicator rule book.
`inv_trans`	Optional inverse transformation function. Will be overridden if specified by the indicator rule book.
`confidence_level`	Numeric between 0 and 1. Confidence level.
`expected_years`	(Optional) Vector of years expected in the output. Used to ensure consistent data frame structure during bootstrapping.
`boot_args`	Named list of additional arguments passed to `bootstrap_cube()`, overriding defaults.
`ci_args`	Named list of additional arguments passed to `calculate_bootstrap_ci()`, overriding defaults.

Details

No computation is performed; the function only returns parameter lists.

Value

A named list with two elements:

bootstrap_params List of parameters for `bootstrap_cube()`

ci_params List of parameters for `calculate_bootstrap_ci()`

Examples

```
ind <- total_occ_ts(example_cube_1, first_year = 2000)
params <- prepare_indicator_bootstrap(
  indicator = ind,
  num_bootstrap = 100,
  ci_type = "perc"
)
names(params$bootstrap_params)
```

print.available_indicators *Print Available Indicators*

Description

Provide a summary of indicators registered in the package.

Usage

```
## S3 method for class 'available_indicators'
print(x, n = 30, ...)
```

Arguments

x	Object of class available_indicators
n	(Optional) Number of indicators to print (default 30).
...	Currently unused.

Value

Invisibly returns the input object x.

print.indicator_map *Print an Indicator Map Object*

Description

Provides a summary representation of an indicator_map object, designed for user-friendly display in the console.

Usage

```
## S3 method for class 'indicator_map'
print(x, n = 10, include_na = FALSE, ...)
```

Arguments

x	An indicator_map object.
n	Integer specifying the number of rows of data to display.
include_na	Logical. If TRUE, includes rows with NA diversity values in the printed output. Default is FALSE.
...	Additional arguments passed to print().

Value

Invisibly returns the input object `x`.

Examples

```
print(example_indicator_map1)
```

---

print.indicator_ts *Print an Indicator Time Series Object*

Description

Provides a summary representation of an indicator_ts object, designed for user-friendly display in the console.

Usage

```
## S3 method for class 'indicator_ts'
print(x, n = 10, ...)
```

Arguments

`x`	An indicator_ts object.
`n`	(Optional) Integer specifying the number of rows of data to display.
`...`	Additional arguments passed to print().

Value

Invisibly returns the input object `x`.

Examples

```
print(example_indicator_ts1)
```

print.processed_cube *Print a Processed Data Cube Object*

Description

Provides a summary representation of a processed_cube object, designed for user-friendly display in the console.

Usage

```
## S3 method for class 'processed_cube'
print(x, n = 10, ...)
```

Arguments

x	A processed_cube object.
n	(Optional) Integer specifying the number of rows of cube data to display.
...	Additional arguments passed to print().

Value

Invisibly returns the input object x.

Examples

```
print(example_cube_1)
```

print.processed_cube_dsinfo *Print a Processed Data Cube Object with Dataset Information*

Description

Provides a summary representation of a processed_cube_dsinfo object, designed for user-friendly display in the console.

Usage

```
## S3 method for class 'processed_cube_dsinfo'
print(x, n = 10, ...)
```

Arguments

x	A processed_cube_dsinfo object.
n	(Optional) Integer specifying the number of rows of data to display.
...	Additional arguments passed to print().

Value

Invisibly returns the input object x.

---

print.sim_cube *Print a Simulated Data Cube Object*

Description

Provides a summary representation of a sim_cube object, designed for user-friendly display in the console.

Usage

```
## S3 method for class 'sim_cube'
print(x, n = 10, ...)
```

Arguments

x	A sim_cube object.
n	(Optional) Integer specifying the number of rows of cube data to display.
...	Additional arguments passed to print().

Value

Invisibly returns the input object x.

---

process_cube *Process GBIF Data Cubes*

Description

Processes a GBIF occurrence cube (a CSV file or a data frame) into a processed_cube object. Cubes produced by the GBIF cube API can have user-specified column names, so check that your column names match the Darwin Core names expected by this function; if not, supply them with the cols_* arguments. The function stops with an error if it cannot find all required columns.

Usage

```
process_cube(
  cube_name,
  grid_type = c("automatic", "eea", "mgrs", "eqdgc", "isea3h", "custom", "none"),
  first_year = NULL,
  last_year = NULL,
  force_gridcode = FALSE,
  cols_year = NULL,
  cols_yearMonth = NULL,
  cols_yearMonthDay = NULL,
  cols_cellCode = NULL,
  cols_occurrences = NULL,
  cols_scientificName = NULL,
  cols_minCoordinateUncertaintyInMeters = NULL,
  cols_minTemporalUncertainty = NULL,
  cols_kingdom = NULL,
  cols_family = NULL,
  cols_species = NULL,
  cols_kingdomKey = NULL,
  cols_familyKey = NULL,
  cols_speciesKey = NULL,
  cols_familyCount = NULL,
  cols_sex = NULL,
  cols_lifeStage = NULL,
  separator = NULL
)
```

Arguments

`cube_name` Either the path to a data cube CSV file (e.g. `system.file("extdata", "denmark_mammals_cube_eqdgc package = "b3gbi")`) or a data frame containing the cube.

`grid_type` (Optional) The grid reference system your cube uses. One of `"automatic"` (default), `"eea"`, `"mgrs"`, `"eqdgc"`, `"isea3h"`, `"custom"` or `"none"`. With `"automatic"` the function attempts to detect the grid from the cell codes and returns an error if it fails. If you want to perform analysis on a cube with custom grid codes (e.g. output from the gcube package) or a cube without grid codes, select `"custom"` or `"none"`, respectively.

`first_year` (Optional) The first year of occurrences to include. If not specified, uses the earliest year present in the cube.

`last_year` (Optional) The final year of occurrences to include. If not specified, uses the latest year present in the cube.

`force_gridcode` (Optional) Logical. If `TRUE`, skips the check that cell codes match the expected format of `grid_type`. Not recommended; invalid codes may cause downstream errors. Default `FALSE`.

`cols_year` (Optional) The name of the column containing the year of occurrence (if something other than 'year'). This column is required unless you have a yearMonth column.

`cols_yearMonth` (Optional) The name of the column containing the year and month of occurrence (if present and if other than 'yearMonth'). Use this only if you do not have a year column. The b3gbi package does not use month data, so the function will convert your yearMonth column to a year column.

`cols_yearMonthDay`
(Optional) The name of the column containing the year, month and day of occurrence (if present and if other than 'yearMonthDay'). Use this only if you do not have year or yearMonth columns. The b3gbi package does not use day or month data, so the function will convert your yearMonthDay column to a year column.

`cols_cellCode` (Optional) The name of the column containing the grid reference codes (if other than 'cellCode'). This column is required.

`cols_occurrences`
(Optional) The name of the column containing the number of occurrences (if other than 'occurrences'). This column is required.

`cols_scientificName`
(Optional) The name of the column containing the scientific name of the species (if other than 'scientificName'). Note that it is not necessary to have both a species column and a scientificName column. One or the other is sufficient.

`cols_minCoordinateUncertaintyInMeters`
(Optional) The name of the column containing the minimum coordinate uncertainty of the occurrences (if other than 'minCoordinateUncertaintyInMeters').

`cols_minTemporalUncertainty`
(Optional) The name of the column containing the minimum temporal uncertainty of the occurrences (if other than 'minTemporalUncertainty').

`cols_kingdom` (Optional) The name of the column containing the kingdom the occurring species belongs to (if other than 'kingdom').

`cols_family` (Optional) The name of the column containing the family the occurring species belongs to (if other than 'family').

`cols_species` (Optional) The name of the column containing the name of the occurring species (if other than 'species'). Note that it is not necessary to have both a species column and a scientificName column. One or the other is sufficient.

`cols_kingdomKey`
(Optional) The name of the column containing the kingdom key of the occurring species (if other than 'kingdomKey').

`cols_familyKey` (Optional) The name of the column containing the family key of the occurring species (if other than 'familyKey').

`cols_speciesKey`
(Optional) The name of the column containing the species key of the occurring species (if other than 'speciesKey'). The column is required, but note that if you have a 'taxonKey' column you can provide it as the speciesKey.

`cols_familyCount`
(Optional) The name of the column containing the occurrence count by family.

`cols_sex` (Optional) The name of the column containing the sex of the observed individuals.

`cols_lifeStage` (Optional) The name of the column containing the life stage of the observed individuals.

`separator` (Optional) The column-separating character in your csv file. This should be automatically recognized, so only specify this if you are having trouble.

Value

An object of class `processed_cube` (or `sim_cube` when `grid_type` is `"custom"` or `"none"`): a list of metadata (years, number of species, grid type, resolution, ...) plus the processed occurrences in the `data` element.

Examples

```
cube_name <- system.file("extdata", "denmark_mammals_cube_eqdgc.csv",
                         package = "b3gbi")
denmark_example_cube <- process_cube(cube_name)
denmark_example_cube
```

`relative_occupancy_map`

Calculate Species Relative Occupancy Over Space or Time

Description

Calculate the relative occupancy of each species - the proportion of grid cells in which it has been recorded - either as a gridded map or as a time series. Three denominator definitions are available via the `occ_type` parameter (see 'Details').

Usage

```
relative_occupancy_map(data, occ_type = 0, ...)

relative_occupancy_ts(data, occ_type = 0, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`occ_type` Integer controlling the occupancy denominator. One of:

- `0` (default) - Total-area occupancy: denominator is all grid cells in the study region (including those with no records).
- `1` - Ever-occupied occupancy: denominator is the number of cells with at least one occurrence (any species) anywhere in the time window.
- `2` - Annual occupancy: for time series, the denominator is the number of cells with at least one occurrence (any species) *in that year*; for maps, the per-year proportion is computed and then averaged across the years in which the species was recorded.

...

Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Relative occupancy:

Relative occupancy quantifies how widely distributed a species is within a study region, expressed as a proportion (0-1). The numerator is always the number of post-aggregation grid cells (`cellid`) in which the species has at least one recorded occurrence. The denominator depends on `occ_type`:

occ_type	Name	Denominator
0	Total-area	All grid cells in region (constant)
1	Ever-occupied	Cells with at least 1 occurrence, any species, full window
2	Annual (TS) / Temporal mean (map)	Cells with at least 1 occurrence in *that year*

Type 0 is the most conservative and comparable across different data cubes because the denominator is fixed by the grid definition. A low value means the species occupies few cells relative to the entire region; however, empty cells cannot be assumed truly unoccupied - they may be under-sampled or simply not yet visited.

Type 1 restricts the denominator to cells where *any* occurrence was recorded across the full time window, conditioning on cells with documented sampling effort. Values will always be $\geq$ the corresponding Type 0 value (since the denominator is smaller). Useful when you wish to compare species occupancy within the subset of cells that have been at least partially surveyed.

Type 2 further restricts the denominator *within each year* (for time series) to cells active in that year. For maps, the annual proportion is computed first and then averaged across the years in which the species was recorded (temporal mean). This is the most dynamic measure, as both numerator and denominator can vary per year, reflecting inter-annual changes in sampling footprint.

Presence-only data caveat:

These cubes contain presence-only data. An empty cell does **not** imply that a species was absent - it may simply reflect lack of sampling in that cell. Types 1 and 2 partially address this by conditioning on cells where at least one occurrence was recorded, but those cells still only document where observers were active, not where species were definitively absent. All three types should be interpreted as *recorded* occupancy, not true occupancy.

Cell aggregation:

When `cell_size` is coarser than the native cube resolution, the data are first aggregated to the coarser grid (internally via `aggregate_data_to_coarser_grid()`). All denominators are computed on the post-aggregation `cellid` (not the original `cellCode`). This ensures the indicator is consistent with the resolution at which it is displayed.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'relative_occupancy' containing:

- **cellid** (map) or **year** (ts): spatial or temporal identifier.
- **cellCode**: grid cell codes (map only; if available).
- **taxonKey**: GBIF taxon keys.
- **scientificName**: Species scientific names.
- **diversity_val**: Relative occupancy value in $[0, 1]$.

Functions

- `relative_occupancy_map()`: Calculate relative occupancy as a gridded map.
- `relative_occupancy_ts()`: Calculate relative occupancy as a time series.

References

Sax, D. F., & Gaines, S. D. (2003). Species diversity: from global decreases to local increases. *Trends in Ecology & Evolution*, 18(11), 561-566.

See Also

`compute_indicator_workflow()`

Other species-based indicators: `spec_occ_map()`, `spec_range_map()`

Examples

```
# Type 0: proportion of all grid cells
ro_map <- relative_occupancy_map(example_cube_1,
  level = "country", region = "Denmark", occ_type = 0
)
plot(ro_map, c(2440728, 4265185))

# Type 1: proportion of ever-occupied cells
ro_map_1 <- relative_occupancy_map(example_cube_1,
  level = "country", region = "Denmark", occ_type = 1
)

# Type 2: temporal mean annual occupancy
ro_map_2 <- relative_occupancy_map(example_cube_1,
  level = "country", region = "Denmark", occ_type = 2
)


# Type 0: proportion of all grid cells (default)
ro_ts <- relative_occupancy_ts(example_cube_1, occ_type = 0)
plot(ro_ts, c(2440728, 4265185))

# Type 1: proportion of ever-occupied cells (constant denominator)
ro_ts_1 <- relative_occupancy_ts(example_cube_1, occ_type = 1)

# Type 2: proportion of cells active that year (varying denominator)
ro_ts_2 <- relative_occupancy_ts(example_cube_1, occ_type = 2)
```

`replace_na` *Replace NA Values in Indicator Objects*

Description

This function takes an `indicator_map` or `indicator_ts` object and replaces all NA values in the `diversity_val` column with zeros. This is useful when NA values represent areas or years with no data that should be treated as zero for mapping or analysis purposes. Only use it when zero is a meaningful value for the indicator: for indicators such as evenness, NA means the value is undefined (e.g. fewer than two species), not zero.

Usage

```
replace_na(x)
```

Arguments

`x`	An `indicator_map` or `indicator_ts` object containing a `diversity_val` column with NA values to replace.

Value

The input indicator object with NA values in `diversity_val` replaced by zero. Stops with an error if there are no NA values.

Examples

```
# Observed richness is NA in cells without occurrences, where zero species
# were observed
or_map_filled <- replace_na(example_indicator_map1)
```

---

spec_occ_map — *Calculate Species Occurrences Over Space or Time*

Description

This function calculates the number of occurrences for individual species over a gridded map or as a time series (see 'Details' for more information).

Usage

```
spec_occ_map(data, ...)

spec_occ_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Species occurrences are considered an essential biodiversity variable (EBV). They are mapped by calculating the total number of occurrences of a given species for each cell. This represents the occurrence frequency distribution, and also indicates the observed species distribution. The number of occurrences can act as a proxy for relative abundance of species with a similar detectability, which is an important aspect of biodiversity although not an indicator when calculated in isolation.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'spec_occ' containing the calculated indicator values and metadata.

Functions

- `spec_occ_map()`:
- `spec_occ_ts()`:

See Also

`compute_indicator_workflow()`, `add_ci()`

Other species-based indicators: `relative_occupancy_map()`, `spec_range_map()`

Examples

```
so_map <- spec_occ_map(example_cube_1,
  level = "country",
  region = "Denmark", include_ocean = FALSE
)
plot(so_map, c(2440728, 4265185))


so_ts <- spec_occ_ts(example_cube_1, first_year = 1985)
plot(so_ts, c(2435767, 2434793))
```

`spec_range_map` *Calculate Species Ranges Over Space or Time*

Description

Calculate the cells occupied by individual species over a gridded map or the change in the number of cells occupied as a time series.

Usage

```
spec_range_map(data, ...)

spec_range_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Maps show presence (1) in each occupied cell; time series give the number of occupied cells per species per year.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'spec_range' containing the calculated indicator values and metadata.

Functions

- `spec_range_map()`:
- `spec_range_ts()`:

See Also

`compute_indicator_workflow()`, `add_ci()`

Other species-based indicators: `relative_occupancy_map()`, `spec_occ_map()`

Examples

```
sr_map <- spec_range_map(example_cube_1,
  level = "country",
  region = "Denmark", include_ocean = FALSE
)
plot(sr_map, c(2440728, 4265185))



sr_ts <- spec_range_ts(example_cube_1, first_year = 1985)
plot(sr_ts, c(2440728, 4265185))
```

---

spec_richness_density_map

Calculate Species Richness Density Over Space or Time

Description

This function calculates the density of unique species over a gridded map or as a time series (see 'Details' for more information).

Usage

```
spec_richness_density_map(data, ...)

spec_richness_density_ts(data, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Density is the number of unique species divided by area (km^2): the area of each grid cell for maps, and the total study-region area for time series. This provides similar information to observed species richness, but is adjusted for area. Like observed richness, bootstrapped confidence intervals are not calculated for this indicator because they are statistically inappropriate at the indicator level.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'spec_richness_density' containing the calculated indicator values and metadata.

Functions

- `spec_richness_density_map()`:
- `spec_richness_density_ts()`:

See Also

compute_indicator_workflow(), add_ci()

Examples

```
srd_map <- spec_richness_density_map(example_cube_1,
  level = "country",
  region = "Denmark"
)
plot(srd_map)


srd_ts <- spec_richness_density_ts(example_cube_1, first_year = 1985)
plot(srd_ts)
```

---

sum_by_taxon *Sum Observations by Taxonomic Rank*

Description

This function takes a processed_cube object and aggregates the observation counts by a specified taxonomic rank.

Usage

```
sum_by_taxon(object, rank)
```

Arguments

object	A processed_cube object containing taxonomic information in object$data.
rank	A character string specifying the taxonomic rank to group by (e.g., "family", "kingdom"). This column must exist in object$data.

Value

A tibble with columns total_observations and the specified rank.

Examples

```
family_sums <- sum_by_taxon(example_cube_1, "family")
head(family_sums)
```

`tax_distinct_map` *Calculate Taxonomic Distinctness Over Space or Time*

Description

This function calculates the taxonomic distinctness index (TDI) over a gridded map or as a time series (see 'Details' for more information).

Usage

```
tax_distinct_map(data, rows = 1, ...)

tax_distinct_ts(data, rows = 1, ...)
```

Arguments

`data` A data cube object (class 'processed_cube').

`rows` Deprecated and ignored. Taxa are now looked up by their GBIF taxon key, so there is no ambiguity to resolve.

`...` Arguments passed on to `compute_indicator_workflow`

`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- `"grid"` (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- `"auto"`: determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as `"10km"` or `"500m"`.

A manually selected cell size must be a whole number multiple of the cube's resolution.

`level` (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

`region` (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

`ne_type` (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

`ne_scale` (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

`output_crs` (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

`first_year` (Optional) Exclude data before this year. (Uses all data in the cube by default.)

`last_year` (Optional) Exclude data after this year. (Uses all data in the cube by default.)

`spherical_geometry` (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

`make_valid` (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

`shapefile_path` (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

`shapefile_crs` (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

`invert` (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

Taxonomic distinctness is an essential biodiversity variable (EBV) that measures the taxonomic relatedness between species, providing a measure of biodiversity that accounts for evolutionary relationships. It is calculated as the Taxonomic Distinctness Index (TDI; Clarke & Warwick, 1999; presence-absence form):

$$\frac{\sum\sum_{i<j}\frac{\omega_{ij}}{L}}{\frac{S(S-1)}{2}}$$

where S is the number of species, ω_{ij} is the taxonomic distance between species i and j (the number of taxonomic levels below the lowest rank they share: 1 for species in the same genus, 2 for the same family, and so on up to 7 for species in different kingdoms), and L = 7 is the number of taxonomic levels used (kingdom, phylum, class, order, family, genus, species). The TDI ranges from 0 to 1, with higher values indicating greater taxonomic distinctness. It is calculated for grid cells or years with at least three classified species.

The classification of each taxon is retrieved from GBIF with the rgbif package (Chamberlain et al.) using the taxon keys in the cube: the GBIF Backbone Taxonomy for numeric keys, and the Catalogue of Life eXtended Release (COL XR) for alphanumeric keys. Taxa are looked up by key in one batched request per checklist; taxa not found by key are retried by scientific name, and taxa still unclassified are excluded (with a warning). Results are cached for the rest of the R session. Requires rgbif (>= 3.7.0; >= 3.8.4 for COL XR keys) and an internet connection.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'tax_distinct' containing the calculated indicator values and metadata.

Functions

- `tax_distinct_map()`:
- `tax_distinct_ts()`:

References

Chamberlain, S., Barve, V., Mcglinn, D., Oldoni, D., Desmet, P., Geffert, L., & Ram, K. rgbif: Interface to the Global Biodiversity Information Facility API. R package. https://CRAN.R-project.org/package=rgbif

Clarke, K. R., & Warwick, R. M. (1999). The taxonomic distinctness measure of biodiversity: weighting of step lengths between hierarchical levels. Marine Ecology Progress Series, 184, 21-29.

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
# Requires the rgbif package and an internet connection (GBIF API)
if (requireNamespace("rgbif", quietly = TRUE)) {
  td_map <- tryCatch(
    tax_distinct_map(example_cube_1,
```

```
      level = "country",
      region = "Denmark"
    ),
    error = function(e) {
      message("GBIF could not be reached: ", conditionMessage(e))
      NULL
    }
  )
  if (!is.null(td_map)) plot(td_map)
}



# Requires the rgbif package and an internet connection (GBIF API)
if (requireNamespace("rgbif", quietly = TRUE)) {
  td_ts <- tryCatch(
    tax_distinct_ts(example_cube_1,
      level = "country",
      region = "Denmark"
    ),
    error = function(e) {
      message("GBIF could not be reached: ", conditionMessage(e))
      NULL
    }
  )
  if (!is.null(td_ts)) plot(td_ts)
}
```

`total_occ_map` *Calculate Total Occurrences Over Space or Time*

Description

This function calculates the total number of species occurrence records over a gridded map or as a time series (see 'Details' for more information).

Usage

```
total_occ_map(data, ...)

total_occ_ts(data, ...)
```

Arguments

`data`	A data cube object (class 'processed_cube').
`...`	Arguments passed on to `compute_indicator_workflow`
	`cell_size` (Optional) Length of grid cell sides, in km or degrees. Only used for maps and for time series that require a grid.

- "grid" (default): use the native resolution of the cube. If this would produce more than about 1 million grid cells over the study area (for degree-based cubes: if the resolution is finer than 1 degree for 'world' or 'continent', or finer than 0.1 degrees otherwise), you are asked to confirm in an interactive session, and the function stops with an error in a non-interactive session.
- "auto": determined automatically. For km-based grids it depends on the area of the study region: 100 km for areas of at least 1 million sq km, 10 km for at least 10,000 sq km, 1 km for at least 100 sq km, and 0.1 km for smaller areas. For degree-based grids it is 1 degree for 'world' or 'continent' and 0.1 degrees otherwise. The automatic size is never smaller than the cube's resolution.
- A number (in the units of the cube's resolution, i.e. km or degrees), or for km-based grids a string such as "10km" or "500m".

A manually selected cell size must be a whole number multiple of the cube's resolution.

level (Optional) Spatial level: 'cube', 'continent', 'country', 'world', 'sovereignty', or 'geounit'. (Default: 'cube')

region (Optional) The region of interest (e.g., "Denmark"). Ignored if level is 'cube' or 'world'. (Default: "Europe")

ne_type (Optional) The type of Natural Earth data to download: 'countries', 'map_units', 'sovereignty', or 'tiny_countries'. This parameter is ignored if level is set to 'cube' or 'world'. (Default: "countries")

ne_scale (Optional) The scale of Natural Earth data to download: 'small' - 110m, 'medium' - 50m, or 'large' - 10m. (Default: "medium")

output_crs (Optional) The CRS you want for your calculated indicator. (Leave blank to let the function choose a default based on grid reference system.)

first_year (Optional) Exclude data before this year. (Uses all data in the cube by default.)

last_year (Optional) Exclude data after this year. (Uses all data in the cube by default.)

spherical_geometry (Optional) If set to FALSE, will temporarily disable spherical geometry while the function runs. Should only be used to solve specific issues. (Default is TRUE).

make_valid (Optional) Calls st_make_valid() from the sf package after creating the grid. Increases processing time but may help if you are getting polygon errors. (Default is FALSE).

shapefile_path (optional) Path of an external shapefile to merge into the workflow. For example, if you want to calculate your indicator for particular features such as protected areas or wetlands.

shapefile_crs (Optional) CRS of a .wkt shapefile. If your shapefile is .wkt and you do NOT use this parameter, the CRS will be assumed to be EPSG:4326 and the coordinates will be read in as lat/long. If your shape is NOT a .wkt the CRS will be determined automatically.

invert (optional) Calculate an indicator over the inverse of the shapefile (e.g. if you have a protected areas shapefile this would calculate an indicator over all non protected areas within your cube). Default is FALSE.

`include_land` (Optional) Include occurrences which fall within the land area. Default is TRUE. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually terrestrial. Grid cells which fall partially on land and partially on ocean will be included even if include_land is FALSE. To exclude terrestrial and/or freshwater taxa, you must manually filter your data cube before calculating your indicator.

`include_ocean` (Optional) Include occurrences which fall outside the land area. Default is TRUE. Set as "buffered_coast" to include a set buffer size around the land area rather than the entire ocean area. Note that this is purely a geographic filter, and does not filter based on whether the occurrence is actually marine. Grid cells which fall partially on land and partially on ocean will be included even if include_ocean is FALSE. To exclude marine taxa, you must manually filter your data cube before calculating your indicator.

`buffer_dist_km` (Optional) The distance to buffer around the land if include_ocean is set to "buffered_coast". Default is 50 km.

`force_grid` (Optional) Forces the calculation of a grid even if this would not normally be part of the pipeline, i.e. for time series. A grid is needed for time series of area-based rarity, Hill diversity and relative occupancy (and for completeness with `gridded_average = TRUE`). This is switched on automatically for these indicators: the wrappers `area_rarity_ts()`, `hill0_ts()`, `hill1_ts()` and `hill2_ts()` already set `force_grid = TRUE`, so do not pass it to them. (Default: FALSE)

Details

The total number of occurrences is calculated by summing the occurrences of all species observed for each cell or year. While not itself an indicator, this variable provides an overview of the comprehensiveness and distribution of data in the cube being analysed, and may be helpful, or even vital, for interpreting the results of calculated indicators.

Value

An S3 object with the classes 'indicator_map' or 'indicator_ts' and 'total_occ' containing the calculated indicator values and metadata.

Functions

- `total_occ_map()`:
- `total_occ_ts()`:

See Also

`compute_indicator_workflow()`, `add_ci()`

Examples

```
to_map <- total_occ_map(example_cube_1,
  level = "country",
  region = "Denmark"
```

```
)
plot(to_map)


to_ts <- total_occ_ts(example_cube_1, first_year = 1985)
plot(to_ts)
```

Index

∗ **datasets**
- available_indicators, 11
- example_cube_1, 30
- example_indicator_map1, 31
- example_indicator_ts1, 32
- example_indicator_ts2, 32

∗ **species-based indicators**
- relative_occupancy_map, 78
- spec_occ_map, 83
- spec_range_map, 86

ab_rarity_map *(area_rarity_map)*, 7
ab_rarity_ts *(area_rarity_map)*, 7
add_ci, 3
add_ci(), *10*, *25*, *29*, *38*, *42*, *46*, *48*, *51*, *55*, *85*, *88*, *91*, *94*, *97*
area_rarity_map, 7
area_rarity_ts *(area_rarity_map)*, 7
area_rarity_ts(), *9*, *21*, *25*, *29*, *36*, *42*, *45*, *48*, *50*, *53*, *80*, *85*, *87*, *90*, *93*, *97*
available_indicators, 11

calc_ci, 12
calc_map *(calc_map.completeness)*, 14
calc_map.completeness, 14
calc_ts *(calc_ts.default)*, 16
calc_ts.default, 16
completeness_map, 19
completeness_ts *(completeness_map)*, 19
compute_indicator_workflow, *8*, *19*, 22, *27*, *35*, *40*, *43*, *46*, *49*, *52*, *79*, *83*, *86*, *89*, *92*, *95*
compute_indicator_workflow(), *10*, *22*, *29*, *38*, *42*, *46*, *48*, *51*, *55*, *81*, *85*, *88*, *91*, *94*, *97*
convert_eqdgc_latlong, 25
create_sf_from_utm, 26
cum_richness_ts, 27

example_cube_1, 30
example_indicator_map1, 31
example_indicator_ts1, 32
example_indicator_ts2, 32

get_observed_years, 33

hill0_map, 33
hill0_ts *(hill0_map)*, 33
hill0_ts(), *9*, *21*, *25*, *29*, *36*, *42*, *45*, *48*, *50*, *53*, *80*, *85*, *87*, *90*, *93*, *97*
hill1_map *(hill0_map)*, 33
hill1_ts *(hill0_map)*, 33
hill1_ts(), *9*, *21*, *25*, *29*, *36*, *42*, *45*, *48*, *50*, *53*, *80*, *85*, *87*, *90*, *93*, *97*
hill2_map *(hill0_map)*, 33
hill2_ts *(hill0_map)*, 33
hill2_ts(), *9*, *21*, *25*, *29*, *36*, *42*, *45*, *48*, *50*, *53*, *80*, *85*, *87*, *90*, *93*, *97*

list_species, 39

mgrs_to_latlong_bbox, 39

newness_map, 40
newness_map(), *25*
newness_ts *(newness_map)*, 40

obs_richness_map, 43
obs_richness_ts *(obs_richness_map)*, 43
occ_density_map, 46
occ_density_ts *(occ_density_map)*, 46
occ_turnover_ts, 49

pielou_evenness_map, 51
pielou_evenness_ts *(pielou_evenness_map)*, 51
plot_map, 55
plot_mv, 59
plot_species_map, 60
plot_species_ts, 64
plot_ts, 67

prepare_indicator_bootstrap, 70
print.available_indicators, 72
print.indicator_map, 72
print.indicator_ts, 73
print.processed_cube, 74
print.processed_cube_dsinfo, 74
print.sim_cube, 75
process_cube, 75

relative_occupancy_map, 78, *85*, *88*
relative_occupancy_map(), *25*
relative_occupancy_ts
(relative_occupancy_map), 78
relative_occupancy_ts(), *25*
replace_na, 82

spec_occ_map, *81*, 83, *88*
spec_occ_ts (spec_occ_map), 83
spec_range_map, *81*, *85*, 86
spec_range_ts (spec_range_map), 86
spec_richness_density_map, 88
spec_richness_density_ts
(spec_richness_density_map), 88
sum_by_taxon, 91

tax_distinct_map, 92
tax_distinct_ts (tax_distinct_map), 92
total_occ_map, 95
total_occ_ts (total_occ_map), 95

williams_evenness_map
(pielou_evenness_map), 51
williams_evenness_ts
(pielou_evenness_map), 51