Package ‘benthos’

October 12, 2022

Type Package
Title Marine Benthic Ecosystem Analysis
Description Preprocessing tools and biodiversity measures (species abundance, species richness, population heterogeneity and sensitivity) for analysing marine benthic data. See Van Loon et al. (2015) <doi:10.1016/j.seares.2015.05.002> for an application of these tools.
Version 1.3-8
Date 2022-08-22
Depends R (>= 3.6.0)
Imports dplyr (>= 0.7.0), lazyeval, readr, utils
Suggests testthat (>= 2.1.0), rmarkdown, knitr, ggplot2, tidyr
VignetteBuilder knitr
LazyData true
Encoding UTF-8
License GPL (>= 3)
RoxygenNote 7.2.1
NeedsCompilation no
Author Dennis Walvoort [aut, cre, cph]
Maintainer Dennis Walvoort <dennis.Walvoort@wur.nl>
Repository CRAN
Date/Publication 2022-08-22 10:10:06 UTC

R topics documented:

benthos-package ............................................................. 2
abundance ........................................................................ 3
ambi .................................................................................. 4
as_accepted ................................................................. 5
bray_curtis ................................................................. 6
benthos-package

Marine Benthic Ecosystem Analysis

Description

benthos provides functions for facilitating the analysis of marine benthos data. Examples are indicators like species abundance, species richness, Margalef’s index of diversity, Shannon’s Entropy, AZTI’s Marine Biotic Index, and the Infaunal Trophic Index (ITI). In addition functions for data pooling, genus-to-species conversion and validation and conversion of species names to those recommended by the World Register of Marine Species are provided.

Details

All functions are designed to work seamlessly with the dplyr-package which implements a grammar for structured data manipulation.

The benthos-package contains functions for estimating various species abundance, species richness, species heterogeneity and species sensitivity measures:
• total abundance (total_abundance)
• abundance (abundance)
• species richness (species_richness)
• Margalef’s index of diversity (margalef)
• Rygg’s index of diversity (rygg)
• Hurlbert’s Expected Number of Species (hurlbert)
• Simpson’s measure of concentration (simpson)
• Hurlbert’s probability of interspecific encounter (PIE) (hpie)
• Shannon’s index or entropy (shannon)
• Hill’s diversity number (hill)
• AZTI Marine Biotic Index (AMBI) (ambi)
• Infaunal Trophic Index (ITI) (iti)
• Bray-Curtis dissimilarity (bray_curtis)

In addition, functions are available for data preparation, e.g.:

• data pooling (pool)
• genus to species conversion (genus_to_species)

For an overview of all the functions in the package click on the index link at the bottom of this page.

Author(s)
Dennis Walvoort <dennis.walvoort@wur.nl>

See Also
The BEQI2-package on CRAN, and the package vignettes.

<table>
<thead>
<tr>
<th>abundance</th>
<th>Abundance</th>
</tr>
</thead>
</table>

Description
The number of individuals in each taxon.

Usage

abundance(.data = NULL, taxon = NULL, count)

abundance_(.data = NULL, taxon = NULL, count)
Arguments

.data          data in a data.frame, tibble, data.table, database etc.
taxon          name of column in .data containing taxa
count          name of column in .data containing counts

Value

numeric vector with abundance per taxon.

Functions

• abundance_(): version suitable for calling from a function (see package lazyeval).

Note

due to pooling, the abundance is not necessarily an integer

Examples

abundance(
    taxon = c("Euspira pulchella", "Nephtys cirrosa" ),
    count = c(4, 6)
)

Description

AZTI Marine Biotic Index (AMBI) according to Borja et al. (2000)

Usage

ambi(.data = NULL, taxon, count, group = NULL)
ambi_(.data = NULL, taxon, count, group = NULL)
has_ambi(.data = NULL, taxon, group = NULL)
has_ambi_(.data = NULL, taxon, group = NULL)

Arguments

.data          data in a data.frame, tibble, data.table, database etc.
taxon          species names
count          counts of individuals (numeric)
group          sensitivity groups I, II, III, IV, or V
Details

The index is given by:

\[ c_b = \frac{3}{2} \sum_{i=2}^{5} (i - 1)p_i \]

where \( p_i \) is the proportion of species in sensitivity group \( i \).

Value

numeric vector of length 1 containing the AMBI

Functions

- \texttt{ambi(\ldots)}: version suitable for calling from a function (see package \texttt{lazyeval}).
- \texttt{has_ambi(\ldots)}: tests if an AMBI sensitivity group is available for \texttt{taxon} (returns \texttt{TRUE} (available) or \texttt{FALSE} (unavailable))
- \texttt{has_ambi(\ldots)}: version suitable for calling from a function (see package \texttt{lazyeval}).

References


Examples

\begin{verbatim}
ambi(
    taxon = c("Euspira pulchella", "Nephtys cirrosa"),
    count = c(4, 6)
)
\end{verbatim}

\begin{verbatim}
data(oosterschelde)
has_ambi(oosterschelde, TAXON)
\end{verbatim}

Description

Taxon names are standardized according to the World Register of Marine Species (WoRMS) database. The conversion is case-insensitive. For this conversion, the TWN-list (Taxa Water management the Netherlands) is used, extended with species of the Southern North Sea. See references below for download locations.

Usage

\begin{verbatim}
as_accepted(taxon, taxa = NULL)
\end{verbatim}
Arguments

- **taxon**: character vector, containing taxon names
- **taxa**: an optional table usually created with `read_taxa`.

Value

character vector with WoRMS/TWN compliant species names

References

- [https://www.marinespecies.org/](https://www.marinespecies.org/)
- [https://taxainfo.nl/](https://taxainfo.nl/)

---

**bray_curtis**  
*Bray-Curtis Dissimilarity*

Description

Bray-Curtis Dissimilarity

Usage

`bray_curtis(n1, n2)`

Arguments

- **n1**: abundances of species at site 1
- **n2**: abundances of species at site 2

Value

Bray-Curtis dissimilarity (0..1, 0 = equal, 1 = different)

Note

species in n1 and n2 need to be aligned

Examples

```r
n1 <- c(11, 0, 7, 8, 0)
n2 <- c(24, 37, 5, 18, 1)
bray_curtis(n1, n2)
```
eqr  

Ecological Quality Ratio (EQR)

Description

The ecological quality ratio is the ratio between a parameter value and its reference value:

\[
EQR = \frac{x - \text{bad}}{\text{ref} - \text{bad}}
\]

Depending on bad and ref, the EQR usually (but not necessarily!) varies between 0 (bad ecological quality) and 1 (ecological quality equals the reference status).

Usage

eqr(x, bad, ref)

Arguments

- x  numeric vector containing benthic indices
- bad  the value for a bad status
- ref  the value for a reference status

Value

numeric vector with EQR-values: low values indicate bad ecological quality and high values indicate good ecological quality.

---

genus_to_species  
Genus to Species Conversion

Description

This algorithm reallocates the counts of taxa, that are only indentified at the genus level to taxa in the same sampling unit and of the same genus but that are identified on the species level. The redistribution of counts is proportional to the number of counts at the species level.

Usage

genus_to_species(is_genus, count)

Arguments

- is_genus  logical vector with elements TRUE if the corresponding taxon is on the genus level, and FALSE if it is on the species level.
- count  numeric vector with elements giving the counts of each corresponding taxon.
Value

numeric vector with updated counts. The counts for the taxon on the genus level have been set to zero.

Note

Parameters is_genus and count are of the same length and correspond to the same taxon. The resulting counts are not necessarily integers.

Examples

genus_to_species(is_genus = c(TRUE, FALSE, FALSE), count = c(3, 10, 20))
genus_to_species(is_genus = c(TRUE, FALSE, FALSE), count = c(1, 10, 20))

get_ambi

Get Supplementary AMBI Sensitivity Groups

Description

This function gets sensitivity groups that are supplementary to the AMBI of Borja et al., (2000)

Usage

get_ambi(which = "NL")

Arguments

which which AMBI supplement? Currently only the Dutch supplement is available (which = "NL")

Value

a data frame with columns TAXON containing taxa and GROUP containing Dutch AMBI-groups

References

get_iti

Get Infaunal Trophic Index

Description
This function gets the sensitivity groups to estimate the infaunal trophic index of Gittenberger et al., (2011)

Usage
get_iti()

Value
a data frame with columns TAXON containing taxa and GROUP containing the ITI-groups of Gittenberger & Van Loon (2013).

References

harmonize

Harmonize Case

Description
Convert text to the most occurring case. In case of ties, the first occurrence in sorted order will be taken.

Usage
harmonize(x)

Arguments
x character vector

Value
character vector with harmonized names (i.e., same case)

Examples
x <- c("FOO", "Foo", "bar", "F00", "bAr", "FO0", "Bar")
y <- harmonize(x)
stopifnot(all.equal(y, c("FO0", "F00", "bar", "F00", "bar", "FO0", "bAr")))
Hill

Description

According to Hill (1973): "a diversity number is figuratively a measure of how many species are present if we examine the sample down to a certain depth among its rarities. If we examine superficially (e.g., by using $N_2$) we shall see only the more abundant species. If we look deeply (e.g., by using $N_0$) we shall see all the species present."

Hill’s diversity numbers are given by:

$$N_a = \sum_i i = 1^S(p_i^a)^{1/(1-a)}$$

Special cases are:

- $N_{-\infty}$ reciprocal of the proportional abundance of the rarest species;
- $N_0$ total number of species present;
- $N_1 \exp(H)$, where $H$: Shannon’s index (see also shannon);
- $N_2$ reciprocal of Simpson’s index (see also simpson);
- $N_{\infty}$ reciprocal of the proportional abundance of the commonest species.

Usage

```r
hill(.data = NULL, taxon, count, a = 0)
hill_(.data = NULL, taxon, count, a = 0)
hill0(.data = NULL, taxon, count)
hill0_(.data = NULL, taxon, count)
hill1(.data = NULL, taxon, count)
hill1_(.data = NULL, taxon, count)
hill2(.data = NULL, taxon, count)
hill2_(.data = NULL, taxon, count)
```

Arguments

- `.data` data in a `data.frame`, `tibble`, `data.table`, database etc.
- `taxon` name of column in `.data` containing taxa
- `count` name of column in `.data` containing counts
- `a` exponent in Hill’s diversity number (R, with special cases for a in 0, 1, 2 (see details))
Value
numeric vector of Hill’s numbers

Functions
- `hill()`: version suitable for calling from a function (see package `lazyeval`).
- `hill0()`: $N_0$
- `hill0_()`: $N_0$, version suitable for calling from a function (see package `lazyeval`).
- `hill1()`: $N_1$
- `hill1_()`: $N_1$, version suitable for calling from a function (see package `lazyeval`).
- `hill2()`: $N_2$
- `hill2_()`: $N_2$, version suitable for calling from a function (see package `lazyeval`).

References

See Also
- `species_richness`, `shannon`, `simpson`

Examples
```r
hill(
  taxon = c("Euspira pulchella", "Nephtys cirrosa"),
  count = c(6, 12),
  a = 0
)
hill0(
  taxon = c("Euspira pulchella", "Nephtys cirrosa"),
  count = c(6, 12)
)
```

---

hpie

Hurlbert’s Probability of Interspecific Encounter (PIE)

Description
The probability that two individuals selected at random (without replacement) from a sample will belong to different species is given by (Hurlbert, 1971, p.579, Eq. 3):

$$
\Delta_1 = \sum_{i=1}^{S} \left( \frac{N_i}{N} \right) \left( \frac{N - N_i}{N - 1} \right) = \left( \frac{N}{N - 1} \right) \Delta_2
$$
where $\Delta_2$ (Hurlbert, 1971, p.579, Eq. 4) is the probability that two individuals selected at random (with replacement) from a sample will belong to different species:

$$\Delta_2 = 1 - \sum_{i=1}^{S} \pi_i^2$$

where $N_i$ is the number of individuals of the $i$th species in the community, $N$ is the total number of individuals in the community, $\pi_i = N_i/N$, and $S$ is the number of species in the community. Note that Hurlbert’s PIE $hpie$ is the complement of $simpson$.

Usage

```r
hpie(.data = NULL, taxon, count)
```

```r
hpie_(.data = NULL, taxon, count)
```

Arguments

- `.data`: data in a data.frame, tibble, data.table, database etc.
- `taxon`: name of column in `.data` containing taxa
- `count`: name of column in `.data` containing counts

Value

A numeric vector with the probability of interspecific encounter (PIE).

Functions

- `hpie_()`: suitable for calling from a function (see package `lazyeval`).

References


See Also

`simpson`, `hurlbert`

Examples

```r
hpie(
    taxon = c("Euspira pulchella", "Nephtys cirrosa"),
    count = c(6, 12)
)
```
hhhurhur blbert

hurlbert  Hurlbert’s Expected Number of Species

Description

The expected number of species in a sample of n individuals:

Usage

hurlbert(.data = NULL, taxon, count, n = 100L)

hurlbert_(.data = NULL, taxon, count, n = 100L)

Arguments

.data data in a data.frame, tibble, data.table, database etc.
taxon name of column in .data containing taxa
count name of column in .data containing counts
n number of individuals in a standard sample

Value

expected number of species in a sample of n individuals

Functions

• hurlbert_(): version suitable for calling from a function (see package lazeval).

References


Examples

hurlbert(
    taxon = c("Euspira pulchella", "Nephtys cirrosa"),
    count = c(4, 6),
    n = 8
)
**isazoic**

**Test for Azoic Samples**

**Description**

Case-insensitive test for taxa starting with ‘azo‘.

**Usage**

```r
is_azoic(x)
```

**Arguments**

- `x` character vector containing taxa

**Details**

Azoic samples need special attention during data analysis. They should be marked as ‘azoic’, and taken care of during analysis. Note that an azoic sample is not the same as a record where a taxon has zero counts. The latter should be removed from further analysis, whereas the former provides important information.

**Value**

Logical vector, with elements TRUE for azoic samples, and FALSE otherwise.

**is_binomen**

**Binomial Names**

is_binomial tests for valid binomial names, generic_name extracts the genus to which the species belongs, specific_name extracts the species within the genus.

**Description**

Binomial Names

is_binomial tests for valid binomial names, generic_name extracts the genus to which the species belongs, specific_name extracts the species within the genus.

**Usage**

```r
is_binomen(x)

generic_name(x)

specific_name(x)

strip_sp(x)
```
Arguments

- **character** vector, containing the binomial name(s) of species (a.k.a. binomen or scientific name)

Value

character vector with either the generic name or the specific name of the species.

Functions

- `generic_name()`: extracts the genus to which the species belongs
- `specific_name()`: extracts the species within the genus
- `strip_sp()`: strips postfix sp. or spp. from a binomen

Examples

```r
is_binomen("Venerupis corrugata") # TRUE
generic_name("Venerupis corrugata") # Venerupis
specific_name("Venerupis corrugata") # corrugata
generic_name("venerupis corrugata") # NA (genus part should be capitalized)
```

---

**iti**  
*Infaunal Trophic Index (ITI)*

Description

Computes the Infaunal Trophic Index (ITI) according to Gittenberger & van Loon (2013).

Usage

```r
iti(.data = NULL, taxon, count, group = NULL)
iti_(.data = NULL, taxon, count, group = NULL)
has_iti(.data = NULL, taxon, group = NULL)
has_iti_(.data = NULL, taxon, group = NULL)
```

Arguments

- `.data` data in a data.frame, tibble, data.table, database etc.
- `taxon` species names
- `count` counts of individuals (numeric)
- `group` sensitivity groups I, II, III, or IV
Details

The Infaunal Trophic Index (ITI) is given by

\[
ITI = 100 \sum_{i=1}^{3} \left( \frac{4 - i}{3} \right) p_i
\]

where \( p_i \) is the proportion of species in class \( i \), where

- group I are suspension feeders (highest quality);
- group II are interface feeders
- group III are surface deposit feeders and
- group IV are subsurface deposit feeders (lowest quality).

Value

numeric vector of length 1 containing the ITI

Functions

- \texttt{iti()}\footnote{\texttt{iti}()}\footnote{version suitable for calling from a function (see package \texttt{lazyeval}).}
- \texttt{has_iti()}\footnote{\texttt{has_iti()}\footnote{tests if an ITI sensitivity group is available for taxon (returns \texttt{TRUE} (available) or \texttt{FALSE} (unavailable))}}
- \texttt{has_iti()}\footnote{\texttt{has_iti()}\footnote{version suitable for calling from a function (see package \texttt{lazyeval}).}}

References


Examples

\[
\texttt{iti(taxon = c("Euspira pulchella", "Nephtys cirrosa"), count = c(4, 6))}
\]

\[
\texttt{data(oosterschelde)}
\]

\[
\texttt{has_iti(oosterschelde, TAXON)}
\]
**margalef**

**Margalef Index of Diversity**

**Description**

Margalef Index of Diversity is given by

\[
D = \frac{S - 1}{\ln(N)}
\]

For \( N = 1 \), the index is set to 0.

**Usage**

\[
\text{margalef(.data = NULL, taxon, count)}
\]

\[
\text{margalef_(.data = NULL, taxon, count)}
\]

**Arguments**

- `.data` : data in a data.frame, tibble, data.table, database etc.
- `taxon` : taxa names (character)
- `count` : counts (numeric)

**Value**

Margalef diversity index (numeric vector of length 1)

**Functions**

- `margalef_()` : version suitable for calling from a function (see package `lazyeval`).

**Examples**

```r
margalef(
  taxon = c("Euspira pulchella", "Nephtys cirrosa"),
  count = c(4, 6)
)
```
**northsea**  
*MWTL North Sea Bentos Data*  

**Description**  
MWTL North Sea Bentos Data  

**Usage**  
northsea  

**Format**  
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 24983 rows and 9 columns.  

---  

**oosterschelde**  
*Oosterschelde Marine Benthos Data*  

**Description**  
Oosterschelde data set. The Oosterschelde is located in the southwest of the Netherlands.  

**Usage**  
oosterschelde  

**Format**  
An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4269 rows and 8 columns.  

**Details**  
The Oosterschelde data contains the following columns:  
- ID sample identifier  
- HABITAT specification of the habitat  
- AREA sampled area  
- DATE sampling date (YYYY-MM-DD, ISO 8601)  
- TAXON standardized taxon code (see WoRMS-website [https://www.marinespecies.org/](https://www.marinespecies.org/))  
- COUNT number of individuals of `TAXON`  

**Note**  
This is not the original data set, but a simplified version of it meant for didactic purposes only! For instance it only contains taxa identified at the species level. Other taxa have been removed.
Description

This function randomly assigns samples to pools of approximately equal area.

Usage

```r
pool(sample_id = 1:length(area), area, target_area, max_try = 100L)
.pool(sample_id = 1:length(area), area, target_area, max_try = 100L)
```

Arguments

- `sample_id` sample identifier
- `area` sampling area of `sample_id` (in the same units as `target_area`)
- `target_area` vector of length 2 containing the lower and upper bound of the pooled area (same units as `area`)
- `max_try` maximum number of unsuccessful pooling tries before the algorithm gives up.

Value

vector with identifiers (integers) indicating the pool to which each sample belongs (NA for samples that could not be pooled).

Functions

- `.pool()`: internal function not supposed to be called directly.
**read_ambi**

**Read and Validate AMBI Sensitivity Data**

**Description**

This function reads and checks files with AMBI sensitivity data. The data should be stored in 'comma separated values' format (csv) consisting of two columns:

- TAXON species name;
- GROUP Roman numeral (I, II, III, IV, V) giving the sensitivity group

**Usage**

```
read_ambi(filename)
validate_ambi(.data)
```

**Arguments**

- `filename` name of the AMBI sensitivity file (character)
- `.data` table in AMBI-format

**Details**

The function performs the following tasks:

- checks the existence of `filename`;
- checks availability of required columns (case insensitive);
- removes redundant spaces;
- removes duplicated records.

**Functions**

- `validate_ambi()`: validator for AMBI-format

**References**

**Description**

This function reads and checks BEQI2 input files. The format has been specified in Van Loon (2013) and is described in the vignette of the BENMMI-package.

**Usage**

```r
read_beqi2(filename)
validate_beqi2(.data)
```

**Arguments**

- `filename`: name of BEQI2 input file (character)
- `.data`: table in BEQI2-format

**Details**

The function performs the following tasks:

- checks the existence of `filename`;
- checks availability of required columns (case insensitive);
- make column names with aggregation data case-insensitive;
- removes redundant spaces;
- checks if DATE-field adheres to ISO 8601 (YYYY-mm-dd);
- constructs a unique identifier `ID` by concatenating columns `OBJECTID` and `DATE`;
- checks that each `ID` has a unique `AREA`;
- checks azeic samples for `VALUE`=0;
- removes records with `VALUE`=0, not belonging to azeic samples;
- checks `VALUE`-field on missing values;
- checks if `VALUE`-field is an integer;

**Functions**

- `validate_beqi2()`: validator for BEQI2-format

**References**

Willem van Loon, 2013. BEQI2 INPUT FORMAT. See the package-vignette of the BENMMI-package.
**read_iti**

*Read and Validate Infaunal Trophic Index Files*

**Description**

This function reads and checks files containing Infaunal Trophic Index (ITI) data (Gittenberger & Van Loon, 2013)

**Usage**

```r
read_iti(filename)
```

```r
validate_iti(.data)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filename</td>
<td>name of the ITI file (character).</td>
</tr>
<tr>
<td>.data</td>
<td>table in ITI-format</td>
</tr>
</tbody>
</table>

**Details**

The function performs the following tasks:

- checks the existence of `filename`;
- checks availability of required columns (case insensitive), i.e., TAXON and GROUP;
- removes redundant spaces;
- removes duplicated records;
- checks if all ITI classes are I, II, III, or IV

The column 'GROUP' contains the Roman numerals I, II, III, and IV, with the following meaning:

- I: suspension feeders;
- II: interface feeders;
- III: surface deposit feeders;
- IV: subsurface deposit feeders.

**Value**

A data frame with columns TAXON containing taxa and GROUP containing user-defined ITI-groups (see Gittenberger & Van Loon, 2013).

**Functions**

- `validate_iti()`: validator for ITI-format
**read_ref**

**References**


---

**read_ref**

*Read and Validate Habitat References Files*

**Description**

This function reads and checks files with reference values.

**Usage**

```r
read_ref(filename, indicators = c("S", "H", "AMBI"))
```

```r
validate_ref(.data, indicators = c("S", "H", "AMBI"))
```

**Arguments**

- `filename`: name of the habitat reference file (character)
- `indicators`: indicators to be processed (character, see details)
- `.data`: table in REF-format

**Details**

The function performs the following tasks:

- checks the existence of `filename`;
- checks availability of required columns (case insensitive);
- removes redundant spaces;
- removes duplicated records.

Argument `indicators` is a character vector of additional benthic indicators to be checked for. For example, if `indicators = "ITI"`, then the habitat reference file should also contain columns ITIREF and ITIBAD. Implemented indicators are N, LNN, S, D, SN, SNA, H, L, AMBI, ITI, PIE, N2 (see package vignette).

The format of the habitat reference file is documented in the BEQI2-package vignette.

**Functions**

- `validate_ref()`: validator for REF-format

**References**

Description

This function reads files in the taxa format.

Usage

read_taxa(filename)

get_taxa()

validate_taxa(.data)

Arguments

filename: name of taxa file
.data: table in taxa-format

Details

Taxa files have the following format:

- group: taxonomic group
- provided: provided taxon name
- accepted: accepted taxon name
- level: taxonomic level

Other columns are allowed, but silently ignored.

Functions

- get_taxa(): get default taxa list (TWN list extended with species Southern North Sea)
- validate_taxa(): validator for taxa-format
read_twn

Read and Validate Taxa Waterbeheer Nederland (TWN) Data

Description
This function reads files in the Taxa Waterbeheer Nederland (TWN) format.

Usage
read_twn(filename)

get_worms()

validate_twn(.data)

Arguments
filename  name of TWN file (character)
.data  table in TWN-format

Details
The function adds a new column taxon. Its contents depending on TWN-status:
- status = 10 taxonname
- status = 20 prefername
- status = 80 parentname

Value
a tibble with four columns:
- GROUP TWN/WoRMS taxon group
- LEVEL TWN/WoRMS taxon level
- FROM taxon name to convert from
- TO taxon name to convert to

Functions
- get_worms(): get default WoRMS list (TWN list extended with species Southern North Sea)
- validate_twn(): validator for TWN-format

References
https://taxainfo.nl/
rygg

Rygg’s Index of Diversity

Description

Rygg’s index of diversity is given by

\[ SN = \frac{\ln(S)}{\ln(\ln(N))} \]

The adjusted version of Rygg’s index which gives more consistent values for smaller \( S=2, N=2, \)
\( N=3 \) and \( S=3, N=3 \) is

\[ SN = \frac{\ln(S)}{\ln(\ln(N+1) + 1)} \]

Usage

\[
\text{rygg(.data = NULL, taxon, count, adjusted = FALSE)}
\]

\[
\text{rygg(.data = NULL, taxon, count, adjusted = FALSE)}
\]

Arguments

- \(.data\)  data in a \texttt{data.frame}, \texttt{tibble}, \texttt{data.table}, database etc.
- \(\text{taxon}\)  taxa names (character)
- \(\text{count}\)  counts (numeric)
- \(\text{adjusted}\)  (defaults to FALSE)

Value

Rygg’s index of diversity (numeric vector of length 1)

Functions

- \(\text{rygg}()\): version suitable for calling from a function (see package \texttt{lazyeval}).

Note

Rygg’s index is not defined for \( N = \exp(1) \). For \( N \leq \exp(1) \), \texttt{rygg} returns \texttt{NA_real_.}

References

**shannon**

**Examples**

```r
rygg(
  taxon = c("Euspira pulchella", "Nephtys cirrosa"),
  count = c(4, 6)
)
```

---

**shannon**  
**Shannon’s Index or Entropy**

**Description**

Compute entropy according to Shannon (1948)

**Usage**

```r
shannon(.data = NULL, taxon, count, base = 2)
```

```r
shannon_(.data = NULL, taxon, count, base = 2)
```

**Arguments**

- `.data` data in a `data.frame, tibble, data.table, database` etc.
- `taxon` taxa names (character)
- `count` counts (numeric)
- `base` the base with respect to which logarithms are computed. Defaults to 2 (unit: bits).

**Value**

Shannon’s entropy

**Functions**

- `shannon_()`: version suitable for calling from a function (see package `lazyeval`).

**References**


**Examples**

```r
shannon(
  taxon = c("Euspira pulchella", "Nephtys cirrosa"),
  count = c(4, 6)
)
```
**simpson**

*Simpson’s Measure of Concentration*

**Description**

The probability that two individuals selected at random (with replacement, Hurlbert, 1971, p.579) from a sample will belong to the same species. For an infinite sample Simpson’s Index is given by (Peet, 1974):

\[
\lambda = \sum_{i=1}^{S} p_i^2
\]

For a finite sample by:

\[
L = \sum_{i=1}^{S} \frac{n_i(n_i - 1)}{N(N - 1)}
\]

where \(p_i\) the proportion of the individuals in species \(i\), \(n_i\) the number of individuals in species \(i\) (relative abundance), and \(N\) the total number of individuals (total abundance). The finite sample case has been implemented in function simpson (and simpson_).

**Usage**

simpson(.data = NULL, taxon, count)

simpson_(.data = NULL, taxon, count)

**Arguments**

- `.data` data in a data.frame, tibble, data.table, database etc.
- `taxon` name of column in .data containing taxa
- `count` name of column in .data containing counts

**Value**

The probability that two individuals selected at random from a sample will belong to the same species.

**Functions**

- `simpson_()`: version suitable for calling from a function (see package `lazyeval`).

**References**


**See Also**

hpie
species_richness

Examples

```r
simpson(
    taxon = c("Euspira pulchella", "Nephtys cirrosa"),
    count = c(6, 12)
)
```

---

species_richness  
Species Richness

Description

Species richness ($S$) is defined as the number of taxa (lowest identification level possible) per sampling unit (data pool or box core sample).

Usage

```r
species_richness(.data = NULL, taxon, count = NULL)
```

```r
species_richness_(.data = NULL, taxon, count = NULL)
```

Arguments

- `.data` data in a `data.frame`, `tibble`, `data.table`, database etc.
- `taxon` taxa names (character)
- `count` number of individuals for each taxon (numeric)

Value

species richness (integer vector of length 1)

Functions

- `species_richness_()`: version suitable for calling from a function (see package `lazyeval`).

Examples

```r
species_richness(
    taxon = c("Euspira pulchella", "Nephtys cirrosa"),
    count = c(4, 6)
)
```
strip_spaces  

Remove Redundant Spaces

Description

This function removes redundant spaces from character vectors.

Usage

`strip_spaces(x)`

Arguments

- `x`: character vector

Value

character vector without trailing or multiple spaces

total_abundance  

Total Abundance

Description

The total number of individuals.

Usage

`total_abundance(.data = NULL, count, na.rm = FALSE)`

`total_abundance_(.data = NULL, count, na.rm = FALSE)`

`lnn(.data = NULL, count, na.rm = FALSE)`

`lnn_(.data = NULL, count, na.rm = FALSE)`

Arguments

- `.data`: data in a data.frame, tibble, data.table, database etc.
- `count`: counts (numeric)
- `na.rm`: Should missing values (including NaN) be removed? (logical)

Value

total number of individuals (integer)
Functions

- `total_abundance_()`: version suitable for calling from a function (see package `lazyeval`).
- `lnn()`: natural log of total abundance + 1 (see package `lazyeval`).
- `lnn_()`: version of lnn suitable for calling from a function (see package `lazyeval`).

Examples

total_abundance(count = c(4, 6))

---

**to_worms**

Convert Taxon Names to Comply with WoRMS

**Description**

Taxon names are standardized according to the World Register of Marine Species (WoRMS) database. The conversion is case-insensitive. For this conversion, the TWN-list (Taxa Water management the Netherlands) is used, extended with species of the Southern North Sea. See references below for download locations.

**Usage**

to_worms(taxon, worms = NULL)

is_worms(.data = NULL, taxon)

is_worms_(.data, taxon)

is_accepted(.data = NULL, taxon)

is_accepted_(.data, taxon)

**Arguments**

- `taxon` character vector, containing taxon names
- `worms` an optional table usually created with `read_twn`
- `.data` data in a `data.frame`, `tibble`, `data.table`, database etc.

**Value**

- character vector with WoRMS compliant species names
- TRUE for WoRMS compliant species names, FALSE otherwise.
- TRUE for WoRMS/TWN compliant species names, FALSE otherwise.
Functions

- `is_worms()`: check if a taxon complies with WoRMS
- `is_worms_()`: as `is_worms` but suitable for calling from a function (see package `lazyeval`).
- `is_accepted()`: check if a taxon complies with WoRMS/TWN
- `is_accepted_()`: as `is_accepted` but suitable for calling from a function (see package `lazyeval`).

References

- [https://www.marinespecies.org/](https://www.marinespecies.org/)
- [https://taxainfo.nl/](https://taxainfo.nl/)
Index

* datasets
  - northsea, 18
  - oosterschelde, 18
  - .pool (pool), 19
  - abundance, 3, 3, 28
  - abundance_ (abundance), 3
  - ambi, 3, 4
  - ambi_ (ambi), 4
  - as_accepted, 5
  - benthos (benthos-package), 2
  - benthos-package, 2
  - bray_curtis, 3, 6
  - character, 6, 15, 31
  - eqr, 7
  - generic_name (is_binomen), 14
  - genus_to_species, 3, 7
  - get_ambi, 8
  - get_iti, 9
  - get_taxa (read_taxa), 24
  - get_worms (read_twn), 25
  - harmonize, 9
  - has_ambi (ambi), 4
  - has_ambi_ (ambi), 4
  - has_iti (iti), 15
  - has_iti_ (iti), 15
  - hill, 3, 10
  - hill0 (hill), 10
  - hill0_ (hill), 10
  - hill1 (hill), 10
  - hill1_ (hill), 10
  - hill2 (hill), 10
  - hill2_ (hill), 10
  - hill_ (hill), 10
  - hpie, 3, 11, 28
  - hpie_ (hpie), 11
  - hurlbert, 3, 12, 13
  - hurlbert_ (hurlbert), 13
  - is_accepted (to_worms), 31
  - is_accepted_ (to_worms), 31
  - is_azoic, 14
  - is_binomen, 14
  - is_worms (to_worms), 31
  - is_worms_ (to_worms), 31
  - iti, 3, 15
  - iti_ (iti), 15
  - lnn (total_abundance), 30
  - lnn_ (total_abundance), 30
  - margalef, 3, 17
  - margalef_ (margalef), 17
  - NA Real, 26
  - northsea, 18
  - oosterschelde, 18
  - pool, 3, 19
  - read_ambi, 20
  - read_beqi2, 21
  - read_iti, 22
  - read_ref, 23
  - read_taxa, 6, 24
  - read_twn, 25, 31
  - rygg, 3, 26
  - rygg_ (rygg), 26
  - shannon, 3, 10, 11, 27
  - shannon_ (shannon), 27
  - simpson, 3, 10–12, 28
  - simpson_ (simpson), 28
  - species_richness, 3, 11, 29
  - species_richness_ (species_richness), 29
  - specific_name (is_binomen), 14
strip_sp(is_binomen), 14
strip_spaces, 30
to_worms, 31
total_abundance, 3, 28, 30
total_abundance_(total_abundance), 30
validate_ambi(read_ambi), 20
validate_beqi2(read_beqi2), 21
validate_iti(read_iti), 22
validate_ref(read_ref), 23
validate_taxa(read_taxa), 24
validate_twn(read_twn), 25