Package 'BioTIMEr'

March 4, 2026

Type Package

Title Tools to Use and Explore the 'BioTIME' Database

Version 0.3.2

Maintainer Alban Sagouis <alban.sagouis@idiv.de>

License MIT + file LICENSE

URL https://biotimehub.github.io/BioTIMEr/

BugReports https://github.com/bioTIMEHub/BioTIMEr/issues

Description The 'BioTIME' database was first published in 2018 and inspired ideas, questions, project and research article. To make it even more accessible, an R package was created.
The 'BioTIMEr' package provides tools designed to interact with the 'BioTIME' database. The functions provided include the 'BioTIME' recommended methods for preparing (gridding and rarefaction) time series data, a selection of standard biodiversity metrics (including species richness, numerical abundance and exponential Shannon) alongside examples on how to display change over time. It also includes a sample subset of both the query and meta data, the full versions of which are freely available on the 'BioTIME' website <https://biotime.st-andrews.ac.uk/home.php>.

Depends R (>= 4.3.0)

Imports tidyr, dplyr, data.table, ggplot2, broom, vegan, dggridR (>= 3.1.0), checkmate, lifecycle

Suggests maps, quarto, knitr, testthat (>= 3.0.0), vdiffr, rlang

Config/testthat/edition 3

Config/testthat/parallel true

Config/testthat/start-first *gridding*, *runResampling*, *metrics*, *workflow*, *slopes*, *plots*, *scales*

Language en-GB

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

VignetteBuilder quarto

NeedsCompilation no

Author Alban Sagouis [aut, cre] (ORCID:
<https://orcid.org/0000-0002-3827-1063>),
Faye Moyes [aut] (ORCID: <https://orcid.org/0000-0001-9687-0593>),
Inês S. Martins [aut, rev] (ORCID:
<https://orcid.org/0000-0003-4328-7286>),
Shane A. Blowes [ctb] (ORCID: <https://orcid.org/0000-0001-6310-3670>),
Viviana Brambilla [ctb] (ORCID:
<https://orcid.org/0000-0002-0560-4693>),
Cher F. Y. Chow [ctb] (ORCID: <https://orcid.org/0000-0002-1020-8409>),
Ada Fontrodona-Eslava [ctb] (ORCID:
<https://orcid.org/0000-0001-7275-7174>),
Laura Antão [ctb, rev] (ORCID: <https://orcid.org/0000-0001-6612-9366>),
Jonathan M. Chase [fnd] (ORCID:
<https://orcid.org/0000-0001-5580-4303>),
Maria Dornelas [fnd, cph] (ORCID:
<https://orcid.org/0000-0003-2077-7055>),
Anne E. Magurran [fnd] (ORCID: <https://orcid.org/0000-0002-0036-2795>),
European Research Council grant AdG BioTIME 250189 [fnd],
European Research Council grant PoC BioCHANGE 727440 [fnd],
European Research Council grant AdG MetaCHANGE 101098020 [fnd],
The Leverhulme Centre for Anthropocene Biodiversity grant RC-2018-021
[fnd],
German Centre for Integrative Biodiversity Research (iDiv)
Halle-Jena-Leipzig [fnd] (ROR: <https://ror.org/01jty7g66>),
Martin Luther University Halle-Wittenberg [fnd] (ROR:
<https://ror.org/05gqaka33>),
University of St Andrews [fnd] (ROR: <https://ror.org/02wn5qz54>)

Repository CRAN

Date/Publication 2026-03-04 10:40:08 UTC

Contents

BTsubset_data *BioTIME subset*

Description

A subset of data from BioTIME temporal surveys.

Usage

```
BTsubset_data
```

Format

'BTsubset_data' A data frame with 81,084 rows and 17 columns:

ID_ALL_RAW_DATA Unique BioTIME identifier for record

ABUNDANCE Double representing the abundance for the record (see metadata for details of ABUNDANCE_TYPE

BIOMASS Double representing the biomass for the record (see metadata for details of BIOMASS_TYPE

ID_SPECIES Unique identifier linking to the species table

SAMPLE_DESC Concatenation of variables comprising unique sampling event

LATITUDE Latitude of record

LONGITUDE Longitude of record

DEPTH Depth or elevation of record if available

DAY Numerical day of record

MONTH Numerical value of month for record, i.e. January=1

YEAR Year of record

STUDY_ID BioTIME study unique identifier

newID Validated species identifier key

valid_name Highest taxonomic resolution of individual, preferred is genus and species

resolution Level of resolution, i.e. 'species' represented by genus and species

taxon Higher level taxonomic grouping, i.e. Fish

Source

<https://biotime.st-andrews.ac.uk/download.php>

BTsubset_meta *BioTIME subset metadata*

Description

A subset of the metadata from BioTIME

Usage

```
BTsubset_meta
```

Format

'BTsubset_meta' A data frame with 12 rows and 25 columns:

STUDY_ID BioTIME study unique identifier

REALM Realm of study location, i.e. Marine

CLIMATE Climate of study location, i.e. Temperate

HABITAT Habitat of study location, i.e. Rivers

PROTECTED_AREA Binary variable indicating if the study is within a protected area

BIOME_MAP Biome of study location (taken from the WWF biomes, i.e. Temperate broadleaf and mixed forests

TAXA High level taxonomic identity of study species, i.e. Fish

ORGANISMS More detailed information on taxonomy, i.e. woody plants

TITLE Title of study as identified in original source

AB_BIO A, B or AB to designate abundance only, biomass only or both

DATA_POINTS Number of unique data points in study, e.g. 10 data points spanning 15 years = 10

START_YEAR First year of study

END_YEAR Last year of study

CENT_LAT Central latitude taken from the convex hull around all study coordinates

CENT_LONG Central longitude taken from the convex hull around all study coordinates

NUMBER_OF_SPECIES Number of distinct species in study

NUMBER_OF_SAMPLES Number of distinct samples in study

NUMBER_LAT_LONG Number of distinct geographic coordinates in study

TOTAL Total number of records in study

GRAIN_SIZE_TEXT Grain size described in text, i.e. size of forest plots

AREA_SQ_KM Total area of study in km2

DATE_STUDY_ADDED Date that the study was added to the database

ABUNDANCE_TYPE Type of abundance, i.e. count

BIOMASS_TYPE Type of biomass, i.e. weight

SAMPLE_DESC_NAME Structure of SAMPLE_DESC

Source

<https://biotime.st-andrews.ac.uk/download.php>

`getAlphaMetrics` *Alpha diversity metrics*

Description

Calculates a set of standard alpha diversity metrics

Usage

```
getAlphaMetrics(x, measure)
```

Arguments

`x`	(`data.frame`) BioTIME data in the format of the output of the `resampling` function. The `resamp` column is optional.
`measure`	(`character`) chosen currency defined by a single column name.

Details

The function `getAlphaMetrics` computes nine alpha diversity metrics for a given community data frame, where `measure` is a character input specifying the abundance or biomass field used for the calculations. For each row of the data frame with data, `getAlphaMetrics` calculates the following metrics:

- Species richness (`S`) as the total number of species in each year with currency > 0.

- Numerical abundance (`N`) as the total currency (sum) in each year (either total abundance or total biomass).

- Maximum Numerical abundance (maxN) as the highest currency value reported in each year.

- Shannon or Shannon–Weaver index is calculated as $\sum_i p_i log_b p_i$, where p_i is the proportional abundance of species i and b is the base of the logarithm (natural logarithms), while exponential Shannon is given by `exp(Shannon)`.

- Simpson's index is calculated as $1 - sum(p_i^2)$, while Inverse Simpson as $1/sum(p_i^2)$.

- McNaughton's Dominance is calculated as the sum of the pi of the two most abundant species.

- Probability of intraspecific encounter or PIE is calculated as $\left(\frac{N}{N-1}\right)\left(1 - \sum_{i=1}^{S} \pi_i^2\right)$.

Note that the input data frame needs to be in the format of the output of the `gridding` function and/or `resampling` functions, which includes keeping the default BioTIME data column names. If such columns are not found an error is issued and the computations are halted. There is an exception for the `resamp` column: the function runs even without it.

Value

Returns a data.frame with results for species richness (S), numerical abundance (N), maximum numerical abundance (maxN), Shannon Index (Shannon), Exponential Shannon (expShannon), Simpson's Index (Simpson), Inverse Simpson (InvSimpson), Probability of intraspecific encounter (PIE) and McNaughton's Dominance (DomMc) for each year and assemblageID.

Examples

```
# Mean and sd values of the metrics for several resamplings
gridding(BTsubset_meta, BTsubset_data) |>
  resampling(measure = "BIOMASS", resamps = 2) |>
  getAlphaMetrics(measure = "BIOMASS") |>
  dplyr::summarise(
     dplyr::across(
        .cols = !resamp, # FIXME
        .fns = c(mean = mean, sd = sd)),
     .by = c(assemblageID, YEAR)) |>
  tidyr::pivot_longer(
     col = dplyr::contains("_"),
     names_to = c("metric", "stat"),
     names_sep = "_",
     names_transform = as.factor) |>
  tidyr::pivot_wider(names_from = stat) |>
  head(10)
```

---

getBetaMetrics *Beta diversity metrics*

Description

Calculates a set of standard beta diversity metrics

Usage

```
getBetaMetrics(x, measure)
```

Arguments

x	(data.frame) BioTIME data table in the format of the output of the resampling functions. The resamp column is optional.
measure	(character) chosen currency defined by a single column name.

Details

The function getBetaMetrics computes three beta diversity metrics for a given community data frame, where `measure` is a character input specifying the abundance or biomass field used for the calculations. `getBetaMetrics` calls the `vegdist` function which calculates for each row the following metrics: Jaccard dissimilarity (`method = "jaccard"`), Morisita-Horn dissimilarity (`method = "horn"`) and Bray-Curtis dissimilarity (`method = "bray"`). Here, the dissimilarity metrics are calculated against the baseline year of each assemblage time series i.e. the first year of each time series. Note that the input data frame needs to be in the format of the output of the `gridding` and/or `resampling` functions, which includes keeping the default BioTIME data column names. If such columns are not found an error is issued and the computations are halted. There is an exception for the `resamp` column: the function runs even without it.

Value

Returns a `data.frame` with results for Jaccard dissimilarity (`JaccardDiss`), Morisita-Horn dissimilarity (`MorisitaHornDiss`), and Bray-Curtis dissimilarity (`BrayCurtsDiss`) for each year and `assemblageID`.

Examples

```
gridding(BTsubset_meta, BTsubset_data) |>
  resampling(measure = "BIOMASS", verbose = FALSE, resamps = 2) |>
  getBetaMetrics(measure = "BIOMASS") |>
  head()
```

---

getLinearRegressions *Get Linear Regressions BioTIME*

Description

Fits linear regression models to `getAlphaMetrics` or `getBetaMetrics` outputs

Usage

```
getLinearRegressions(x, pThreshold = 0.05)
```

Arguments

`x`	(`data.frame`) BioTIME data table in the format of the output of `getAlphaMetrics` or `getBetaMetrics` functions
`pThreshold`	(`numeric`) P-value threshold for statistical significance

Details

The function `getLinearRegression` fits simple linear regression models (see `lm` for details) for a given output ('data') of either `getAlphaMetrics` or `getBetaMetrics` function. The typical model has the form `metric ~ year`. Note that assemblages with less than 3 time points and/or single species time series are removed.

Value

Returns a single long data.frame with results of linear regressions (slope, p-value, significance, intercept) for each assemblageID.

Examples

```
x <- gridding(BTsubset_meta, BTsubset_data) |>
resampling(measure = "BIOMASS", verbose = FALSE, resamps = 2)

  alpham <- getAlphaMetrics(x, "BIOMASS")

getLinearRegressions(x = alpham, pThreshold = 0.01) |> head(10)

  betam <- getBetaMetrics(x = x, "BIOMASS")

getLinearRegressions(x = betam) |> head(10)
```

---

gridding *gridding BioTIME data*

Description

grids BioTIME data into a discrete global grid based on the location of the samples (latitude/longitude).

Usage

```
gridding(meta, btf, res = 12, resByData = FALSE, verbose = TRUE)
```

Arguments

meta	(data.frame, tibble or data.table) BioTIME metadata.
btf	(data.frame, tibble or data.table) BioTIME data.
res	(integer) cell resolution. Must be in the range [0,30]. Larger values represent finer resolutions. Default: 12 (~96 sq km). Passed to dgconstruct.
resByData	(logical) FALSE by default. If TRUE, the function dg_closest_res_to_area is called to adapt res to the data extent. The new res value is used even if a value is provided byt the user.
verbose	if TRUE, a warning will be shown when one-year-long time series are found in btf and excluded.

Details

Each BioTIME study contains distinct samples which were collected with a consistent methodology over time, and each with unique coordinates and date. These samples can be fixed plots (i.e. SL or 'single-location' studies where measures are taken from a set of specific georeferenced sites at any given time) or wide-ranging surveys, transects, tows, and so on (i.e. ML or 'multi-location' studies where measures are taken from multiple sampling locations over large extents that may or may not align from year to year, see `runResampling`. `gridding` is a function designed to deal with the issue of varying spatial extent between studies by using a global grid of hexagonal cells derived from `dgconstruct` and assigning the individual samples to the cells across the grid based on its latitude and longitude. Specifically, each sample is assigned a different combination of study ID and grid cell resulting in a unique identifier for each assemblage time series within each cell (assemblageID). This allows for the integrity of each study and each sample to be maintained, while large extent studies are split into local time series at the grid cell level. By default meta represents a long form data frame containing the data information for BioTIME studies and `btf` is a data frame containing long form data from a main BioTIME query (see Example). `res` defines the global grid cell resolution, thus determining the size of the cells (see `vignette("dggridR")`). `res = 12` was found to be the most appropriate value when working on the whole BioTIME database(corresponding to ~96 km2 cell area), but the user can define their own grid resolution (e.g. `res = 14`, or when `resbyData = TRUE` allow the function to find the best `res` based on the average study extent.

Value

Returns a `'data.frame'`, with selected columns from the `btf` and `meta` data frames, an extra integer column called `'cell'` and two character columns called 'StudyMethod' and 'assemblageID' (concatenation of `STUDY_ID` and `cell`).

Examples

```
## Not run:
  gridded_data <- gridding(meta = BTsubset_meta, btf = BTsubset_data)
  gridded_data <- gridding(meta = dplyr::as_tibble(BTsubset_meta),
                           btf = dplyr::as_tibble(BTsubset_data))
  gridded_data <- gridding(meta = data.table::as.data.table(BTsubset_meta),
                           btf = data.table::as.data.table(BTsubset_data))

## End(Not run)
```

---

resampling *Rarefy BioTIME data to an equal number of samples per year*

Description

Takes the output of `gridding` and applies sample-based rarefaction to standardise the number of samples per year within each cell-level time series (i.e. assemblageID).

Usage

```
resampling(
  x,
  measure,
  resamps = 1L,
  conservative = FALSE,
  summarise = TRUE,
  verbose = TRUE
)
```

Arguments

x	(data.frame) BioTIME gridded data to be resampled (in the format of the output of the gridding function).
measure	(character) currency to be retained during the sample-based rarefaction. Can be either defined by a single column name or a vector of two or more column names.
resamps	(integer) number of repetitions. Default is 1.
conservative	(logical). FALSE by default. If TRUE, whenever a NA is found in the measure field(s), the whole sample is removed instead of the missing observations only.
summarise	(logical). TRUE by default. If FALSE, the function returns abundance and/or biomass summed at the SAMPLE_DESC level git pull(i.e., per sample), rather than per species per year.
verbose	(logical). TRUE by default. If FALSE, warnings when NA values or one-year-long time series are found in x and excluded are hidden.

Details

Sample-based rarefaction prevents temporal variation in sampling effort from affecting diversity estimates (see Gotelli N.J., Colwell R.K. 2001 Quantifying biodiversity: procedures and pitfalls in the measurement and comparison of species richness. Ecology Letters 4(4), 379-391) by selecting an equal number of samples across all years in a time series. resampling counts the number of unique samples taken in each year (sampling effort), identifies the minimum number of samples across all years, and then uses this minimum to randomly resample each year down to that number. Thus, standardising the sampling effort between years, standard biodiversity metrics can be calculated based on an equal number of samples (e.g. using getAlphaMetrics, getAlphaMetrics). measure is a character input specifying the chosen currency to be used during the sample-based rarefaction. It can be a single column name or a vector of two or more column names - e.g. for BioTIME, measure="ABUNDANCE", measure="BIOMASS" or measure = c("ABUNDANCE", "BIOMASS").

By default, any observations with NA within the currency field(s) are removed. You can choose to remove the full sample where such observations are present by setting conservative to TRUE. resamps can be used to define multiple iterations, effectively creating multiple alternative datasets as in each iteration different samples will be randomly selected for the years where number of samples > minimum. Note that the function always returns a single data frame, i.e. if resamps > 1, the returned data frame is the result of individual data frames concatenated together, one from each iteration identified by a numerical unique identifier 1:resamps.

Value

Returns a single long form `data.frame` containing the total currency or currencies of interest (sum) for each species in each year within each rarefied time series (i.e. `assemblageID`). An extra integer column called `resamp` indicates the specific iteration.

Examples

```
## Not run:
  set.seed(42)
  x <- gridding(BTsubset_meta, BTsubset_data)
  resampling(x, measure = "BIOMASS", summarise = TRUE)
  resampling(x, measure = "ABUNDANCE", verbose = FALSE)
  resampling(x, measure = c("ABUNDANCE","BIOMASS"))
  # Without summarising the species abundances are summed at the SAMPLE_DESC level
  resampling(x, measure = "BIOMASS", summarise = FALSE, conservative = FALSE)

## End(Not run)
```

---

scale_color_biotime *Scale construction for ggplot use*

Description

Scale construction for ggplot use

Scale construction for filling in ggplot

Usage

```
scale_color_biotime(palette = "realms", discrete = TRUE, reverse = FALSE, ...)

scale_colour_biotime(palette = "realms", discrete = TRUE, reverse = FALSE, ...)

scale_fill_biotime(palette = "realms", discrete = TRUE, reverse = FALSE, ...)
```

Arguments

`palette`	One of: ‘realms‘, ‘gradient‘, ‘cool‘, ‘warm‘, default to ‘realms‘.
`discrete`	See Details. default to ‘FALSE‘
`reverse`	Default to ‘FALSE‘
`...`	Passed to `discrete_scale` or `scale_color_gradient`

Details

USAGE NOTE: Remember to change these arguments when plotting colours continuously.

Value

If `discrete` is `TRUE`, the function returns a colour palette produced by `discrete_scale` and if `discrete` is `FALSE`, the function returns a colour palette produced by `scale_color_gradient`.

If `discrete` is `TRUE`, the function returns a colour palette produced by `discrete_scale` and if `discrete` is `FALSE`, the function returns a colour palette produced by `scale_color_gradient`.

Author(s)

Cher F. Y. Chow

`themeBioTIME` *ggplot2 theme for BioTIME plots*

Description

ggplot2 theme for BioTIME plots

Usage

```
themeBioTIME(
  legend.position,
  font.size,
  axis.colour,
  strip.background,
  axis.color = axis.colour,
  fontSize = deprecated(),
  colx = deprecated(),
  coly = deprecated(),
  lp = deprecated()
)
```

Arguments

`legend.position`	the default position of legends ("none", "left", "right", "bottom", "top", "inside")
`font.size`	Size of axes labels, legend text and title (+1), and title (+2).
`axis.colour`	Colour name for the axes, ticks and axis labels.
`strip.background`	Colour name. Passed to `theme` as `fill` colour for the `strip.background` element.
`axis.color`	US spelling for `axis.colour`.
`fontSize`	Deprecated in Favour of font.size
`colx`	Deprecated in favour of `axis.colour`.
`coly`	Deprecated in favour of `strip.background`.
`lp`	Deprecated in favour of `legend.position`.

Examples

```
## Not run:
  fig1 <- ggplot2::ggplot() +
         themeBioTIME(legend.position = "none", font.size = 12,
           axis.colour = "black", strip.background = "grey90")

## End(Not run)
```

Index