Package ‘letsR’

March 27, 2026

Type Package

Title Data Handling and Analysis in Macroecology

Version 6.0

Date 2026-03-27

Description Handling, processing, and analyzing geographic
data on species' distributions and environmental variables.
Read Vilela & Villalobos (2015) <doi:10.1111/2041-210X.12401> for details.

License GPL-2

Imports geosphere, sf

Depends R (>= 3.1.0), terra, grDevices, graphics, methods, stats

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

LazyData true

URL `https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.12401`,
`https://github.com/macroecology/letsR`,
`https://brunovilela.github.io/letsR/`

BugReports `https://github.com/macroecology/letsR/issues`

RoxygenNote 7.3.3

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation no

Author Bruno Vilela [aut, cre] (ORCID:
<`https://orcid.org/0000-0003-4072-0558`>),
Fabricio Villalobos [aut] (ORCID:
<`https://orcid.org/0000-0002-5230-2217`>)

Maintainer Bruno Vilela <`bvilela.bv@gmail.com`>

Repository CRAN

Date/Publication 2026-03-27 20:50:02 UTC

Contents

`letsR-package` *Tools for Data Handling and Analysis in Macroecology.*

Description

The letsR package is being developed to help researchers in the handling, processing, and analysis of macroecological data. Its purpose is to integrate these methodological processes into a single software platform for macroecological analyses. The package's main functions allow users to build presence-absence matrices, the basic analytical tool in macroecology, from species' geographical distributions and merge them with species' traits, conservation information (downloadable using functions from this package) and spatial environmental layers. In addition, other package's functions enable users to summarize and visualize information from presence-absence matrices.

Details

All functions in this package use a prefix and a suffix separated by a dot. The prefix refers to the package's name and the suffix to the actual function. This is done to avoid confusion with potentially similarly-named functions from other R packages. For instance, the letsR function used to create presence-absence matrices is called `lets.presab` (but see also `lets.presab.birds` and `lets.presab.points`) whereas the one used to add variables to a presence-absence matrix is called `lets.addvar`. The package's basic functions create and work on a particular S3 object class called `PresenceAbsence`. Such `PresenceAbsence` object class allows storing information beyond presence-absence data (e.g. user-defined grid-cell system) and using the generic `plot`, `summary` and `print` functions of R. Also, some package's functions allow the user to input customary R objects (e.g. `vector`, `matrix`, `data.frame`).

If you are looking for the most recent version of the package, you can get the development version of letsR on github (`https://github.com/macroecology/letsR`).

Package:	lestR
Type:	Package
Version:	3.1
Date:	2018-01-24
License:	GPL-2

Author(s)

Bruno Vilela
(email: <`bvilela@wustl.edu`>; Website: `https://bvilela.weebly.com/`)

Fabricio Villalobos
(email: <`fabricio.villalobos@gmail.com`>; Website: `https://fabro.github.io`)

References

Vilela, B., & Villalobos, F. (2015). letsR: a new R package for data handling and analysis in macroecology. Methods in Ecology and Evolution.

`IUCN` *IUCN evaluation for frogs of the genus Phyllomedusa*

Description

Result of the function lets.iucn (deprecated) applied to South American frog genus Phyllomedusa in 2014.

Usage

```
IUCN
```

Format

A data frame with 32 rows and 7 columns:

Species Scientific name

Family Family

Status Red List Status

Criteria Criteria for listing as threatened

Population Population trend per IUCN

Description_Year Year described

Country Presence in country

Source

IUCN - `https://www.iucnredlist.org/`. 2014.

`lets.addpoly` *Add polygon coverage to a PresenceAbscence object*

Description

Add polygon coverage within cells of a PresenceAbsence object.

Usage

```
lets.addpoly(x, y, z, onlyvar = FALSE, count = FALSE)
```

Arguments

x	A PresenceAbsence object.
y	Polygon of interest.
z	A character indicating the column name of the polygon containing the attributes to be used.
onlyvar	If TRUE only the matrix object will be returned.
count	Logical, if TRUE a progress bar indicating the processing progress will be shown.

Value

The result is a presence-absence matrix of species with the polygons' attributes used added as columns at the right-end of the matrix. The Values represent the percentage of the cell covered by the polygon attribute used.

Author(s)

Bruno Vilela

See Also

lets.presab.birds

lets.presab

lets.addvar

Examples

```
## Not run:
data(PAM)  # Phyllomedusa presence-absence matrix
data(wrld_simpl)  # World map
Brazil <- wrld_simpl[wrld_simpl$NAME == "Brazil", ]  # Brazil (polygon)

# Check where is the variable name
# (in this case it is in "NAME" which will be my z value)
names(Brazil)

PAM_pol <- lets.addpoly(PAM, Brazil, "NAME", onlyvar = TRUE)

## End(Not run)
```

`lets.addvar` *Add variables (in raster format) to a PresenceAbscence object*

Description

Add variables (in raster format), usually environmental, to a PresenceAbsence object. Variables are included as additional columns containing the aggregate/summarize value of the variable(s) in each cell of the presence-absence matrix.

Usage

```
lets.addvar(x, y, onlyvar = FALSE, fun = mean)
```

Arguments

`x`	A `PresenceAbsence` object.
`y`	Variables to be added in `SpatRaster` format.
`onlyvar`	If `TRUE` only the `matrix` object will be returned.
`fun`	Function used to aggregate the variables(s) values over each cell. Note that this will only work for variables with a resolution value smaller (i.e. higher resolution) than the PAM.

Value

The result is a presence-absence matrix of species with the variables added as columns at the right-end of the matrix (but see the 'onlyvar' argument).

Note

The `PresenceAbsence` and the `Raster` variable must be in the same projection.

Author(s)

Bruno Vilela

See Also

`lets.presab.birds`

`lets.presab`

`lets.addpoly`

Examples

```
## Not run:
data(temp)  # Global mean temperature
temp <- terra::unwrap(temp)
data(PAM)  # Phyllomedusa presence-absence matrix
# Mean temperature
PAM_temp_mean <- lets.addvar(PAM, temp)
# Standard deviation of temperature
PAM_temp_sd <- lets.addvar(PAM, temp, fun = sd, onlyvar = TRUE)
# Mean and SD in the PAM
PAM_temp_mean_sd <- cbind(PAM_temp_mean, PAM_temp_sd)

## End(Not run)
```

`lets.attrcells` *Descriptors of position, centrality, and isolation in attribute space*

Description

Computes a suite of descriptor variables for each cell of an *attribute-space* presence–absence matrix, as returned by `lets.attrpam`. Attribute variables are treated as a two-dimensional space, and the function derives metrics that characterize: (i) the position of each attribute cell relative to the attribute-space centroid (mean and frequency-weighted distances), (ii) its proximity to attribute-space borders (zero-richness frontier, quantified via multiple distance-based proxies), and (iii) its isolation within attribute space (frequency-weighted Euclidean distance to other cells).

When a geographic presence–absence matrix is supplied, the function also links each attribute cell to the geographic cells occupied by the taxa occurring in that cell, allowing the calculation of: (iv) frequency in geographic space, (v) total geographic area, and (vi) geographic isolation statistics (summaries of pairwise distances among associated geographic cells).

Usage

```
lets.attrcells(x, y = NULL, perc = 0.1, remove.cells = FALSE)
```

Arguments

`x` A list produced by `lets.attrpam` containing, at minimum:

- `$PAM_attribute`: a data frame in which the first column is the attribute-cell identifier (`Cell_attr`), the second and third columns are the coordinates of the two attribute axes, and the remaining columns are taxa coded as presence (1) or absence (0).
- `$Attr_Richness_Raster`: a SpatRaster containing the richness of taxa in each attribute cell.

`y`	A geographic presence–absence object produced by `lets.presab`. If supplied, the function calculates the number of geographic cells associated with each attribute cell, the total area of those cells, and summary statistics of pairwise geographic distances among them. If `NULL`, these geographic descriptors are not calculated, and attribute-cell richness is used as the weighting variable for the attribute-space metrics.
`perc`	Numeric value in the interval 0 to 1 indicating the proportion of the shortest distances to empty attribute cells to be averaged in the robust border-distance metric. Default is `0.1`.
`remove.cells`	Logical. If `TRUE`, removes attribute cells that were not originally present in `x$PAM_attribute` and that were added internally to complete the raster support.

Details

Summarize metrics of the attribute-space PAM

The two attribute variables are standardized to zero mean and unit variance before distance-based calculations.

If `y` is provided, the function first identifies the taxa present in each attribute cell, then retrieves the geographic cells occupied by those taxa in the geographic PAM. Based on these linked geographic cells, the function computes:

- `Frequency`: number of associated geographic cells;
- `Area`: total area of the associated geographic cells;
- summary statistics of pairwise geographic distances among those cells.

If `y = NULL`, geographic descriptors are not computed. In this case, attribute-cell richness is used as the weighting variable in the midpoint and frequency-weighted distance calculations.

Empty attribute cells are defined as cells with zero frequency when `y` is provided, or zero richness when `y = NULL`. Cells with `NA` values in the richness raster are treated as empty.

Distances to the weighted and unweighted midpoints are returned as negative values so that larger values indicate greater centrality in attribute space.

Value

A `data.frame` with one row per attribute cell. The output always contains:

- `Cell_attr`: attribute-cell identifier.
- `Weighted Mean Distance to midpoint`: negative Euclidean distance from the cell to the weighted midpoint of occupied attribute space.
- `Mean Distance to midpoint`: negative Euclidean distance from the cell to the unweighted midpoint of occupied attribute space.
- `Minimum Zero Distance`: minimum distance from the cell to any empty attribute cell.
- `Minimum X% Zero Distance`: mean distance from the cell to the nearest fraction of empty attribute cells defined by `perc`, where `X = perc * 100`.

- `Distance to MCP border`: distance from the cell to the border of the minimum convex polygon enclosing occupied attribute cells.
- `Frequency Weighted Distance`: weighted mean distance from the cell to all other attribute cells.

When `y` is provided, the output additionally includes:

- `Frequency`: number of geographic cells associated with the taxa present in the attribute cell.
- `Area`: summed area of those associated geographic cells.
- `Isolation (Min.)`, `Isolation (1st Qu.)`, `Isolation (Median)`, `Isolation (Mean)`, `Isolation (3rd Qu.)`, and `Isolation (Max.)`: summary statistics of pairwise geographic distances among associated geographic cells.

Examples

```
## Not run:
# Example using a geographic PAM and simulated attribute data
data("PAM")

n <- length(PAM$Species_name)
Species <- PAM$Species_name
trait_a <- rnorm(n)
trait_b <- trait_a * 0.2 + rnorm(n)
df <- data.frame(Species, trait_a, trait_b)

# Build the attribute-space PAM
x <- lets.attrpam(df, n_bins = 4)

# Calculate attribute-cell descriptors using the geographic PAM
cell_desc <- lets.attrcells(x, y = PAM)

# Plot the resulting descriptors
lets.plot.attrcells(x, cell_desc)

## End(Not run)
```

`lets.attrpam` *Attribute-space Presence–Absence Matrix (attrPAM)*

Description

Builds a presence–absence matrix (PAM) in a two-dimensional trait space, by binning species occurrences along two quantitative attributes (e.g., body size and mass). Each species can have one or multiple entries in the trait dataset.

Usage

```
lets.attrpam(
  x,
  n_bins = 10,
  remove.cells = TRUE,
  remove.sp = TRUE,
  count = FALSE
)
```

Arguments

`x`	A data frame where the first column contains species (character vector), and the next two columns contain numeric trait values (2D space).
`n_bins`	Integer. Number of bins per axis (default = 10).
`remove.cells`	Logical. Should cells with no species be removed from the final matrix?
`remove.sp`	Logical. Should species with no occurrences in attribute space be removed?
`count`	Logical. If ‘TRUE‘, displays a progress bar for species processing.

Details

Create a Presence–Absence Matrix in Trait Space

The two trait axes are divided into equal-interval bins, generating a grid of ‘n_bins × n_bins‘ cells. Each species occurrence is assigned to a cell, and the resulting PAM indicates which species are present in each trait cell.

Value

A list with two components:

- `PAM_attribute`: a matrix with cell ID, trait coordinates, and species presence (0/1).
- `Attr_Richness_Raster`: a raster of richness (number of species) in trait space.

Examples

```
## Not run:
n <- 2000
Species <- paste0("sp", 1:n)
trait_a <- rnorm(n)
trait_b <- trait_a * .2 + rnorm(n)
x <- data.frame(Species, trait_a, trait_b)
test <- lets.attrpam(x, n_bins = 30)
lets.plot.attrpam(test)

## End(Not run)
```

`lets.classvar` *Frequency distribution of a variable within a species' range*

Description

Based on a species Presence-Absence matrix including variables of interest (see `lets.addvar`), the function divides a continuous variable into classes and counts the frequency of each class within each species' range.

Usage

```
lets.classvar(x, groups = "default", pos, xy)
```

Arguments

`x`	Presence-absence `matrix` with a single variable added (see `lets.addvar`).
`groups`	The number of classes into which the variable will be divided. Default calculates the number of classes as the default for a histogram (`hist`).
`pos`	Column number containing the variable of interest.
`xy`	Logical, if `TRUE` the input matrix contains the geographic coordinates in the first two columns.

Value

A `matrix` with species in the rows and the variable's classes in the columns.

Author(s)

Bruno Vilela

References

Morales-Castilla et al. 2013. Range size patterns of New World oscine passerines (Aves): insights from differences among migratory and sedentary clades. Journal of Biogeography, 40, 2261-2273.

Examples

```
## Not run:
data(PAM)
data(temp)
pamvar <- lets.addvar(PAM, temp)
resu <- lets.classvar(x = pamvar, pos = ncol(pamvar), xy = TRUE)

## End(Not run)
```

`lets.correl` *Compute correlogram based on the Moran's I index*

Description

Computes the Moran's I correlogram of a single or multiple variables.

Usage

```
lets.correl(x, y, z, equidistant = FALSE, plot = TRUE)
```

Arguments

`x`	A single numeric variable in vector format or multiple variables in matrix format (as columns).
`y`	A distance matrix of class `matrix` or `dist`.
`z`	The number of distance classes to use in the correlogram.
`equidistant`	Logical, if `TRUE` the classes will be equidistant. If `FALSE` the classes will have equal number of observations.
`plot`	Logical, if `TRUE` the correlogram will be ploted.

Value

Returns a matrix with the Moran's I Observed value, Confidence Interval (95 and Expected value. Also the p value of the randomization test, the mean distance between classes, and the number of observations. quase tudo

Author(s)

Bruno Vilela, Fabricio Villalobos, Lucas Jardim & Jose Alexandre Diniz-Filho

References

Sokal, R.R. & Oden, N.L. (1978) Spatial autocorrelation in biology. 1. Methodology. Biological Journal of the Linnean Society, 10, 199-228.

Sokal, R.R. & Oden, N.L. (1978) Spatial autocorrelation in biology. 2. Some biological implications and four applications of evolutionary and ecological interest. Biological Journal of the Linnean Society, 10, 229-249.

Examples

```
## Not run:
data(PAM)
data(IUCN)

# Spatial autocorrelation in description year (species level)
midpoint <- lets.midpoint(PAM)
```

```
distan <- lets.distmat(midpoint[, 2:3])
moran <- lets.correl(IUCN$Description, distan, 12,
                     equidistant = FALSE,
                     plot = TRUE)


## End(Not run)
```

lets.distmat *Compute a geographic distance matrix*

Description

Calculates a geographic distance matrix based on a PresenceAbsence or a two column matrix of x(longitude) and y(latitude).

Usage

```
lets.distmat(xy, asdist = TRUE)
```

Arguments

xy	A PresenceAbsence object or a matrix with two columns (longitude, latitude).
asdist	Logical, if TRUE the result will be an object of class dist, if FALSE the result will be an object of class matrix.

Details

This function basically facilitates the use of terra::distance on a PresenceAbsence object, allowing also the user to have directly a dist object. The distance is always expressed in meter if the coordinate reference system is longitude/latitude, and in map units otherwise. Map units are typically meter, but inspect crs(x) if in doubt.

Value

The user can choose between dist and matrix class object to be returned. The resulting values are in kilometres (but see the argument 'unit' in rdist.earth).

Author(s)

Bruno Vilela & Fabricio Villalobos

Examples

```
## Not run:
data(PAM)
distPAM <- lets.distmat(PAM)

## End(Not run)
```

`lets.envcells` *Environmental descriptors for Presence–Absence in environmental space*

Description

Computes a suite of descriptors for each environmental cell in an *environmental* presence–absence matrix (envPAM), including: (i) frequency in geographic space, (ii) geographic isolation statistics (summary of pairwise distances among geographic cells mapped to the same environmental cell), (iii) distance to environmental midpoints (mean and frequency-weighted mean), (iv) distance to environmental "border" (zero-richness frontier via three proxies), and (v) an environmental isolation metric based on frequency-weighted Euclidean distance in standardized environmental space.

Distances to midpoints are returned inverted (i.e., larger values imply greater centrality in environmental space), following the current implementation.

Usage

```
lets.envcells(x, perc = 0.2, remove.cells = FALSE)
```

Arguments

`x`	A list returned by `lets.envpam` containing at least: • `$Presence_and_Absence_Matrix_env`: data.frame with columns `cell_id_env`, the environmental coordinates (e.g., two variables), and species presences (one column per species). • `$Presence_and_Absence_Matrix_geo`: data.frame with columns `cell_id_geo`, geographic coordinates (lon, lat), and species presences.+ • `$Env_Richness_Raster`: a **terra** SpatRaster of richness in environmental space.
`perc`	Numeric in (0,1], the fraction used in the robust border metric (mean of the *n* smallest distances to zero-richness cells). Default = 0.1.
`remove.cells`	Logical. If 'TRUE', remove empty cells from the results.

Details

Summarize environmental–geographical metrics from an environmental space PAM

Environmental variables (assumed to be the 2nd and 3rd columns of `$Presence_and_Absence_Matrix_env`) are z-scored before computing distances. Geographic isolation is summarized with `summary()` of pairwise distances among geographic cells that collapse to the same environmental cell.

Value

A data frame with one row per environmental cell. The output includes:

- `Cell_env`: Environmental cell identifier.
- `Frequency`: Number of geographic cells mapped to the environmental cell.
- `Area`: Summed area of the associated geographic cells.
- `Isolation (Min.)`, `Isolation (1st Qu.)`, `Isolation (Median)`, `Isolation (Mean)`, `Isolation (3rd Qu.)`, and `Isolation (Max.)`: Summary statistics of pairwise geographic distances among associated geographic cells.
- `Weighted Mean Distance to midpoint`: Negated distance from the cell to the frequency-weighted midpoint in standardized environmental space.
- `Mean Distance to midpoint`: Negated distance from the cell to the unweighted midpoint in standardized environmental space.
- `Minimum Zero Distance`: Minimum distance from the cell to any empty environmental cell.
- A column named according to `perc` (e.g., `Minimum 10% Zero Distance`): Mean distance from the cell to the nearest fraction of empty environmental cells.
- `Distance to MCP border`: Distance from the cell to the environmental-space border.
- `Frequency Weighted Distance`: Frequency-weighted mean distance from the cell to all other environmental cells.

Caveats

(1) The code assumes the first column of `$Presence_and_Absence_Matrix_env` indexes environmental cells and columns 2–3 are the two environmental variables. (2) The geographic matrix assumes lon/lat at columns 3–4. Adjust indices if needed. (3) If there are no zero-richness cells or < 3 occupied env cells, border metrics are returned as `NA`.

See Also

`lets.envpam`, `lets.plot.envcells`, `lets.plot.envpam`

Examples

```
## Not run:
data("Phyllomedusa"); data("prec"); data("temp")
prec <- unwrap(prec); temp <- unwrap(temp)
PAM  <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Precipitation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, terra::vect(wrld_simpl))
res <- lets.envpam(PAM, envs, n_bins = 30)
out <- lets.envcells(res, perc = 0.2)
lets.plot.envcells(res, out)

## End(Not run)
```

lets.envpam *Create a presence–absence matrix in environmental space*

Description

Transform a presence–absence matrix (PAM) based on geographic coordinates into a new PAM structured in environmental space. The function rasterizes the environmental variable space (based on two continuous environmental predictors), and assigns species presences to binned environmental conditions, producing a species richness raster in environmental space.

Usage

```
lets.envpam(
  pam,
  envs,
  n_bins = 30,
  remove.cells = TRUE,
  remove.sp = TRUE,
  count = FALSE
)
```

Arguments

`pam`	A ‘PresenceAbsence‘ object, typically created using `lets.presab`.
`envs`	A two-column matrix or data frame with continuous environmental variables corresponding to the coordinates in the PAM. The first column will be used as the x-axis and the second as the y-axis in the environmental space.
`n_bins`	Number of bins used to discretize each environmental variable. Default is 30.
`remove.cells`	Logical. Should cells with no species be removed from the final matrix?
`remove.sp`	Logical. Should species with no occurrences in environmental space be removed?
`count`	Logical. If TRUE, displays a progress bar for species processing.

Details

This function projects species occurrences into a two-dimensional environmental space, facilitating ecological analyses that depend on environmental gradients. The environmental space is discretized into a regular grid (determined by `n_bins`), and each cell is assigned the number of species occurring under those environmental conditions.

Value

A list with the following elements:

- `Presence_and_Absence_Matrix_env`: A matrix of species presences across environmental bins.

- `Presence_and_Absence_Matrix_geo`: The original PAM coordinates associated with environmental cells.
- `Env_Richness_Raster`: A raster layer of species richness in environmental space.
- `Geo_Richness_Raster`: The original species richness raster in geographic space.

Author(s)

Bruno Vilela

See Also

`lets.presab`, `lets.addvar`, `lets.plot.envpam`

Examples

```
## Not run:
# Load data
data("Phyllomedusa")
data("prec")
data("temp")

prec <- unwrap(prec)
temp <- unwrap(temp)
PAM <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Preciptation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, vect(wrld_simpl))

# Run function
res <- lets.envpam(PAM, envs)

# Plot results
lets.plot.envpam(x = res,
            species = NULL,
            cell_id_env = NULL,
            cell_id_geo = NULL,
            T,
            world = TRUE,
            mar = c(4, 4, 4, 4))

## End(Not run)
```

`lets.field` *Create species' values based on the species co-occurrence within focal ranges*

Description

Create single species' values based on the attributes of species co-occurring within individual ranges.

Usage

```
lets.field(x, y, z, weight = TRUE, xy = NULL, count = FALSE)
```

Arguments

`x`	A `PresenceAbsence` object or a presence-absence in `matrix` format (see `xy` argument for matrix use) with the species named in the columns.
`y`	Species attribute to be considered. It must be a numeric attribute.
`z`	Species names in the same order as the attributes and exactly the same as named in the `matrix` or in the `PresenceAbsence` object.
`weight`	If `TRUE` the value is weighted by species' range size, if `FALSE` the value is the mean of all species that co-occur within the focal species.
`xy`	If `TRUE` the presence-absence `matrix` contains the cell coordinates in the first two columns.
`count`	Logical, if `TRUE` a counting window will open.

Details

If the species do not co-occur with any other species NaN will be returned.

Author(s)

Bruno Vilela & Fabricio Villalobos

References

Villalobos, F. and Arita, H.T. 2010. The diversity field of New World leaf-nosed bats (Phyllostomidae). Global Ecology and Biogeography. 19, 200-211.

Villalobos, F., Rangel, T.F., and Diniz-Filho, J.A.F. 2013. Phylogenetic fields of species: cross-species patterns of phylogenetic structure and geographical coexistence. Proceedings of the Royal Society B. 280, 20122570.

See Also

`lets.presab.birds`

`lets.presab`

Examples

```
## Not run:
data(PAM)
range <- lets.rangesize(x = PAM, units = "cell")
field <- lets.field(PAM, range, PAM$S, weight = TRUE)

## End(Not run)
```

`lets.gridirizer` *Fits a grid into a PresenceAbsence object*

Description

This function creates a grid in shapefile format and adds its cells' IDs to the presence-absence matrix. The function was created to facilitate the use of the PresenceAbsence object for the ones who prefer to work with a grid in shapefile format.

Usage

```
lets.gridirizer(x)
```

Arguments

`x`	A `PresenceAbsence` object.

Value

The result is a list of two objects. The first is a grid in shapefile format; the second is a presence-absence matrix with an aditional column called SP_ID (shapefile cell identifier).

Author(s)

Bruno Vilela

See Also

`plot.PresenceAbsence`

`lets.presab.birds`

Examples

```
## Not run:
data(PAM)
PAM.grid <- lets.gridirizer(PAM)
names(PAM.grid)
# Grid in polygon format (can be saved in shapefile)
PAM.grid$Grid
```

```
# Presence-absence matrix (beggining only)
head(PAM.grid$Presence[, 1:5])


## End(Not run)
```

`lets.iucncont` *Transform IUCN RedList conservation status to continuous values*

Description

Transform IUCN RedList conservation status to continuous values ranging from 0 to 5.

Usage

```
lets.iucncont(x, dd = NA, ne = NA)
```

Arguments

`x`	A vector or a matrix containing IUCN codes to be transformed.
`dd`	The value to be attributed to DD (data-deficient) species, the default option is NA.
`ne`	The value to be attributed to NE (not-evaluated) species, the default option is NA.

Value

Returns a vector/matrix with continuos values from 0 to 5.

EX and EW = 5

CR = 4

EN = 3

VU = 2

NT = 1

LC = 0

DD = NA

NE = NA

Author(s)

Bruno Vilela

References

Purvis A et al., 2000. Predicting extinction risk in declining species. Proceedings of the Royal Society of London. Series B: Biological Sciences, 267.1456: 1947-1952.

Examples

```
## Not run:
#Vector transformation
status <- sample(c("EN","VU", "NT", "CR", "DD", "LC", "EX"),
                 30, replace = TRUE)
transV <- lets.iucncont(status)

#matrix transformation
data(IUCN)
transM <- lets.iucncont(IUCN)


## End(Not run)
```

`lets.load` *Load a PresenceAbsence object*

Description

Reload PresenceAbsence objects written with the function `lets.save`.

Usage

```
lets.load(file)
```

Arguments

`file`	a character string giving the name of the file to load.

Author(s)

Bruno Vilela

See Also

`lets.save`

`lets.presab`

Examples

```
## Not run:
data(PAM)
lets.save(PAM, file = "PAM.RData")
PAM <- lets.load(file = "PAM.RData")

## End(Not run)
```

`lets.maplizer` *Create a matrix summarizing species' attributes within cells of a PresenceAbsence object*

Description

Summarize species atributes per cell in a presence-absence matrix.

Usage

```
lets.maplizer(x, y, z, func = mean, ras = FALSE, weighted = FALSE)
```

Arguments

`x`	A `PresenceAbsence` object.
`y`	Species attribute to be considered. It must be a numeric attribute.
`z`	Species names in the same order as the attributes and exactly the same as named in the `PresenceAbsence` object.
`func`	A function to summarize the species' atribute in each cell (the function must return only one value).
`ras`	If `TRUE` the raster object will be returned together with the matrix.
`weighted`	If TRUE, argument func is ignored, and weighted mean is calculated. Weights are attributed to each species according to 1/N cells that the species occur.

Value

The result can be both a `matrix` or a `list` cointaining the follow objects:

Matrix: a `matrix` object with the cells' geographic coordinates and the summarized species' attributes within them.

Raster: The summarized species'attributed maped in a `SpatRaster` object.

Author(s)

Bruno Vilela

See Also

`lets.presab`

`lets.presab.birds`

Examples

```
## Not run:
data(PAM)
data(IUCN)
trait <- IUCN$Description_Year
resu <- lets.maplizer(PAM, trait, PAM$S, ras = TRUE)
head(resu$Matrix)
plot(resu$Raster, xlab = "Longitude", ylab = "Latitude",
main = "Mean description year per site")


## End(Not run)
```

`lets.maplizer.attr` *Map species-level attributes over the attribute-space grid*

Description

Summarizes species attributes (e.g., trait values, description year) within each attribute-space cell of a presence–absence matrix produced by `lets.attrpam`. A summary function is applied across the attributes of species present in each cell, producing a per-cell attribute surface and an optional raster for visualization.

Usage

```
lets.maplizer.attr(x, y, z, func = mean, ras = TRUE, weighted = FALSE)
```

Arguments

`x`	An object returned by `lets.attrpam`, containing the attribute-space PAM (matrix) and its raster.
`y`	A numeric vector with species attributes to summarize (one value per species in `z`). If `y` is a factor, it is coerced to numeric by level codes.
`z`	A character vector of species names corresponding to `y`. These names must match the species columns in the attribute-space PAM within `x`.
`func`	A function to summarize attributes across species within each cell (e.g., `mean`, `median`, `sum`). Must return a single numeric value.
`ras`	Logical; if `TRUE`, include the attribute map as a SpatRaster in the output.
`weighted`	If TRUE, argument func is ignored, and weighted mean is calculated. Weights are attributed to each species according to 1/N cells that the species occur.

Details

Map species attributes in attribute space

Internally, the function multiplies each species presence column by its attribute value in `y` (matched by name via `z`), sets zeros to `NA` so that summary functions ignore absences, applies `func` across species for each cell, and rasterizes the resulting per-cell summaries onto the attribute raster template. Cells with no species remain as `NA`.

Value

A `list` with:

- `Matrix_attr`: matrix/data.frame with summarized attribute values per attribute cell (first columns are the cell ID and the two attribute axes).
- `Attr_Raster`: SpatRaster with the summarized attribute mapped in attribute space.

Author(s)

Bruno Vilela

See Also

`lets.attrpam`, `lets.plot.attrpam`, `lets.attrcells`

Examples

```
## Not run:
# Simulate a dataset of 2000 species with two traits
n <- 2000
Species <- paste0("sp", 1:n)
trait_a <- rnorm(n)
trait_b <- trait_a * 0.2 + rnorm(n)
df <- data.frame(Species, trait_a, trait_b)

# Build attribute-space PAM
x <- lets.attrpam(df, n_bins = 30)

# Map species-level attribute (here, trait_b) by cell using the mean
res <- lets.maplizer.attr(x, y = trait_b, z = Species)

# Plot attribute raster
lets.plot.attrpam(res)

## End(Not run)
```

`lets.maplizer.env` *Map species attributes in both environmental and geographic spaces*

Description

Summarize species-level attributes (e.g., traits or conservation data) within each cell of environmental and geographic presence–absence matrices, enabling trait-based mapping across environmental gradients and geographic space.

Usage

```
lets.maplizer.env(x, y, z, func = mean, ras = TRUE, weighted = FALSE)
```

Arguments

`x`	An object produced by `lets.envpam`.
`y`	A numeric vector containing the species attributes to be summarized (e.g., description year, body size).
`z`	A character vector with species names corresponding to the values in `y`. These names must match the species columns in the presence-absence matrices in ‘x‘.
`func`	A function used to summarize the attribute across species in each cell (e.g., `mean`, `median`, `sum`). Must return a single numeric value.
`ras`	Logical. If `TRUE`, the result includes the attribute maps as `SpatRaster` objects.
`weighted`	If TRUE, argument func is ignored, and weighted mean is calculated. Weights are attributed to each species according to 1/N cells that the species occur.

Details

This function is useful for trait-based macroecological analyses that aim to understand how species attributes vary across environments or space. It uses the output of `lets.envpam`, applies a summary function to the trait values of all species present in each cell, and returns raster layers for visualization.

Value

A list with the following elements:

- `Matrix_env`: A matrix with summarized attribute values in environmental space.
- `Matrix_geo`: A matrix with summarized attribute values in geographic space.
- `Env_Raster`: A `SpatRaster` with the attribute values mapped in environmental space.
- `Geo_Raster`: A `SpatRaster` with the attribute values mapped in geographic space.

Author(s)

Bruno Vilela

See Also

lets.envpam, lets.maplizer, lets.plot.envpam

Examples

```
## Not run:
# Load data
data("Phyllomedusa")
data("prec")
data("temp")
data("IUCN")

prec <- unwrap(prec)
temp <- unwrap(temp)
PAM <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Preciptation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, vect(wrld_simpl))

# Create environmental PAM
res <- lets.envpam(PAM, envs, remove.cells = FALSE)

# Map mean description year
res_map <- lets.maplizer.env(res,
                             y = IUCN$Description_Year,
                             z = IUCN$Species)

# Plotting trait maps
lets.plot.envpam(res_map)

## End(Not run)
```

---

lets.midpoint *Compute the midpoint of species' geographic ranges*

Description

Calculate species distribution midpoint from a presence-absence matrix using several methods.

Usage

```
lets.midpoint(pam, planar = FALSE, method = "PC", inside = FALSE)
```

Arguments

pam	A presence-absence `matrix` (sites in the rows and species in the columns, with the first two columns containing the longitudinal and latitudinal coordinates, respectively), or an object of class PresenceAbsence.

`planar`	Logical, if `FALSE` the coordinates are in Longitude/Latitude. If `TRUE` the coordinates are planar.
`method`	Default option, "PC" (polygon centroid) will generate a polygon from the raster, and calculate the centroid of this polygon based on the function `terra::centroids`. Note that for the "PC" method, users can only use PresenceAbsence objects. Note also that this method will not be the best for PresenceAbsence objects made from occurrence records, or that have multiple disjoint distributions. Users can also choose the geographic midpoint, using the option "GM". "GM" will create a bounding box across the extremes of the distribution and calculate the centroid. Alternatively, the midpoint can be calculated as the point that minimize the distance between all cells of the PAM, using the method "CMD"(centre of minimum distance). The user can also calculate the midpoint, based on the centroid of the minimum convex polygon of the distribution, using the method "MCC". This last method is useful when using a PresenceAbsence object made from occurrence records.
`inside`	logical. If TRUE the points returned are guaranteed to be inside the polygons or on the lines, but they are not the true centroids. True centroids may be outside a polygon, for example when a polygon is "bean shaped", and they are unlikely to be on their line

Value

A `data.frame` containing the species' names and geographic coordinates (longitude [x], latitude [y]) of species' midpoints.

Author(s)

Fabricio Villalobos & Bruno Vilela

See Also

`lets.presab`

`lets.presab.birds`

Examples

```
## Not run:
data(PAM)
mid <- lets.midpoint(PAM, method = "PC")
mid2 <- lets.midpoint(PAM, method = "GM")
mid3 <- lets.midpoint(PAM, method = "CMD")
mid4 <- lets.midpoint(PAM, method = "MCC")
mid5 <- lets.midpoint(PAM, method = "PC", planar = TRUE)
mid6 <- lets.midpoint(PAM, method = "GM", planar = TRUE)
mid7 <- lets.midpoint(PAM, method = "CMD", planar = TRUE)
mid8 <- lets.midpoint(PAM, method = "MCC", planar = TRUE)

for (sp in seq_len(nrow(mid))) {
 #sp = 4 # Or choose a line or species
 plot(PAM, name = mid[sp, 1])
 points(mid[sp, -1], col = adjustcolor("blue", .8), pch = 20, cex = 1.5)
 points(mid2[sp, -1], col = adjustcolor("green", .8), pch = 20, cex = 1.5)
 points(mid3[sp, -1], col = adjustcolor("yellow", .8), pch = 20, cex = 1.5)
 points(mid4[sp, -1], col = adjustcolor("purple", .8), pch = 20, cex = 1.5)
 points(mid5[sp, -1], col = adjustcolor("orange", .8), pch = 20, cex = 1.5)
 points(mid6[sp, -1], col = adjustcolor("black", .8), pch = 20, cex = 1.5)
 points(mid7[sp, -1], col = adjustcolor("gray", .8), pch = 20, cex = 1.5)
 points(mid8[sp, -1], col = adjustcolor("brown", .8), pch = 20, cex = 1.5)
 Sys.sleep(1)
}

## End(Not run)
```

lets.overlap *Compute pairwise species' geographic overlaps*

Description

Creates a species geographic overlap matrix from a Presence-absence matrix.

Usage

```
lets.overlap(pam, method = "Chesser&Zink", xy = NULL)
```

Arguments

pam	A presence-absence matrix (sites in rows and species in columns, with the first two columns containing the longitudinal and latitudinal coordinates, respectively), or an object of class PresenceAbsence.
method	The method used to calculate the overlap matrix. "Chesser&Zink" calculates the degree of overlap as the proportion of the smaller range that overlaps within the larger range (Chesser & Zink 1994). "Proportional" calculates the proportion of a range that overlaps another range, the resultant matrix is not symmetric. "Cells" will show the number of overlapping grid cells between a pair of species' ranges (same for both species in a pair), here the resultant matrix is symmetric.
xy	Logical, if TRUE the input matrix contains geographic coordinates in the first two columns.

Author(s)

Fabricio Villalobos & Bruno Vilela

References

Chesser, R. Terry, and Robert M. Zink. "Modes of speciation in birds: a test of Lynch's method." Evolution (1994): 490-497.

Barraclough, Timothy G., and Alfried P. Vogler. "Detecting the geographical pattern of speciation from species-level phylogenies." The American Naturalist 155.4 (2000): 419-434.

See Also

lets.presab

lets.presab.birds

Examples

```
## Not run:
data(PAM)
CZ <- lets.overlap(PAM, method = "Chesser&Zink")
prop <- lets.overlap(PAM, method = "Proportional")
cells <- lets.overlap(PAM, method = "Cells")

## End(Not run)
```

---

lets.pamcrop *Crop a PresenceAbsence object based on an input shapefile*

Description

Crop a PresenceAbsence object based on a shapefile provided by the user.

Usage

```
lets.pamcrop(x, shp, remove.sp = TRUE, remove.cells = FALSE)
```

Arguments

`x`	A `PresenceAbsence` object.
`shp`	Object of class SpatVector (see function `terra::vect`) to crop the PresenceAbsence object.
`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`remove.cells`	Logical, if `FALSE` the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).

Value

The result is an object of class PresenceAbsence croped.

Author(s)

Bruno Vilela

See Also

plot.PresenceAbsence

lets.presab.birds

Examples

```
## Not run:
data(PAM)
data("wrld_simpl")

# PAM before crop
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")

# Crop PAM to Brazil
data(wrld_simpl)  # World map
Brazil <- wrld_simpl[wrld_simpl$NAME == "Brazil", ]  # Brazil (polygon)
PAM_crop <- lets.pamcrop(PAM, Brazil, remove.sp = TRUE)
plot(PAM_crop, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness (Brazil crop)")
plot(sf::st_geometry(wrld_simpl), add = TRUE)

## End(Not run)
```

`lets.plot.attrcells` *Map AttrPAM descriptor layers*

Description

Maps each descriptor column returned by `lets.attrcells` back onto the attribute raster. Optionally returns the rasters without plotting.

Usage

```
lets.plot.attrcells(
  x,
  y,
  ras = FALSE,
  plot_ras = TRUE,
  col_func = NULL,
  mfrow = c(4, 4)
)
```

Arguments

`x`	A list returned by `lets.attrpam` (must contain `$Attr_Richness_Raster` and `$PAM_attribute`).
`y`	A `data.frame` returned by `lets.attrcells`, with one row per attribute cell (aligned with `x`).
`ras`	Logical; if `TRUE`, returns a named list of SpatRaster layers for each descriptor column (default `FALSE`).
`plot_ras`	Logical; if `TRUE`, plots each raster (default `TRUE`).

`col_func`	A custom color ramp palette function to use for plotting variables (e.g., from `colorRampPalette`).
`mfrow`	A vector of the form c(nr, nc). The figures will be drawn in an nr-by-nc array on the device by rows as in par documentation.

Details

Plot attribute-cell descriptors as rasters

Rows with zero or `NA` richness are masked before plotting, to avoid edge artifacts from empty attribute cells. The plotting grid defaults to `par(mfrow = c(4, 4))`; adjust as needed.

Value

If `ras = TRUE`, returns a named `list` of SpatRaster layers (one per descriptor column).

Examples

```
## Not run:
# Example with simulated traits
data(PAM)
n <- length(PAM$Species_name)
Species <- PAM$Species_name
trait_a <- rnorm(n)
trait_b <- trait_a * 0.2 + rnorm(n)  # correlated trait
df <- data.frame(Species, trait_a, trait_b)

# Build AttrPAM
x <- lets.attrpam(df, n_bins = 4)

# Compute descriptors
desc <- lets.attrcells(x, PAM)

# Plot descriptors
lets.plot.attrcells(x, desc)

## End(Not run)
```

`lets.plot.attrpam` *Plot attrPAM results*

Description

Plots the richness (or a single species) in the attribute (trait) space generated by `lets.attrpam`.

Usage

```
lets.plot.attrpam(x, species = NULL, col_rich = NULL, ...)
```

Arguments

`x`	A list returned by `lets.attrpam`.
`species`	Optional. A character string with the name of a species to highlight. If provided, plots its presence in trait space.
`col_rich`	Optional. A color palette function or vector. If ‘NULL‘, a default palette is used (red ramp for species maps, red–white ramp for richness).
`...`	Additional arguments passed to `plot()`.

Details

Plot an Attribute-space PAM

Examples

```
## Not run:
# Example with simulated traits
n <- 2000
Species <- paste0("sp", 1:n)
trait_a <- rnorm(n)
trait_b <- trait_a * 0.2 + rnorm(n)  # correlated trait
df <- data.frame(Species, trait_a, trait_b)

# Build AttrPAM
x <- lets.attrpam(df, n_bins = 30)
lets.plot.attrpam(x)

## End(Not run)
```

`lets.plot.envcells` *Map envPAM descriptors*

Description

Plots each column of the descriptor table returned by `lets.envcells` back onto the environmental richness raster embedded in the environmental space PAM object.

Usage

```
lets.plot.envcells(
  x,
  y,
  ras = FALSE,
  plot_ras = TRUE,
  mfrow = c(4, 4),
  which.plot = NULL,
  col_func = NULL,
  ...
)
```

Arguments

`x`	The envPAM list returned by `lets.envpam` (must include `$Env_Richness_Raster` and `$Presence_and_Absence_Matrix_env`).
`y`	A `data.frame` returned by `lets.envcells` with one row per environmental cell (aligned with `x`).
`ras`	Logical; if `TRUE`, returns a named list of **terra** SpatRaster layers corresponding to each column plotted. Default `FALSE`.
`plot_ras`	Logical; if `TRUE`, the function plot the graphs. Default `TRUE`.
`mfrow`	A vector of the form c(nr, nc). The figures will be drawn in an nr-by-nc array on the device by rows as in par documentation.
`which.plot`	Indicate the number of the columns in y to be ploted.
`col_func`	A custom color ramp palette function to use for plotting variables (e.g., from `colorRampPalette`).
`...`	other arguments passed to `terra::plot` function.

Details

Plot environmental descriptors over the environmental raster grid

Each descriptor column is assigned as values of the environmental raster template and plotted sequentially. Rows with zero frequency are masked as `NA`. The plotting grid defaults to `par(mfrow = c(4,4))`.

Value

If `ras = TRUE`, returns a named list of SpatRaster objects corresponding to each descriptor column.

Examples

```
## Not run:
data("Phyllomedusa"); data("prec"); data("temp")
prec <- unwrap(prec); temp <- unwrap(temp)
PAM  <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Precipitation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, terra::vect(wrld_simpl))
res <- lets.envpam(PAM, envs, n_bins = 30)
out <- lets.envcells(res, perc = 0.2)
lets.plot.envcells(res, out)

## End(Not run)
```

lets.plot.envpam *Plot species richness in environmental and geographical space*

Description

This function plots species richness in both environmental and geographical space based on the output of `lets.envpam`. It can optionally highlight species distributions, individual cells, or regions in both spaces.

Usage

```
lets.plot.envpam(
  x,
  species = NULL,
  cell_id_env = NULL,
  cell_id_geo = NULL,
  geo_plot = TRUE,
  env_plot = TRUE,
  world = TRUE,
  rast_return = FALSE,
  col_rich = NULL,
  ...
)
```

Arguments

`x`	The output object from `lets.envpam`).
`species`	A character string indicating the species name to be highlighted in both plots.
`cell_id_env`	An integer or vector of integers indicating environmental space cell(s) to be highlighted.
`cell_id_geo`	An integer or vector of integers indicating geographic cell(s) to be highlighted.
`geo_plot`	Logical. Should the geographic richness map also be plotted? Default is TRUE.
`env_plot`	Logical. Should the environmental space richness map also be plotted? Default is TRUE.
`world`	Logical. If TRUE, plots a base map using the ‘wrld_simpl‘ object from the ‘letsR‘ package over the geographic raster.
`rast_return`	Logical. If TRUE, returns the modified raster objects instead of plotting.
`col_rich`	A custom color ramp palette function to use for plotting richness (e.g., from `colorRampPalette`).
`...`	Additional arguments passed to the `plot` function for rasters.

Details

This function provides a visual summary of species richness across both geographic and environmental dimensions. Users can highlight specific species or environmental/geographical cells. When a highlight is selected, both rasters are modified to display only presences related to the selected species or cells, and all other cells are greyed out.

Author(s)

Bruno Vilela

See Also

`lets.envpam`, `lets.presab`, `plot.PresenceAbsence`

Examples

```
## Not run:
# Load data
data("Phyllomedusa")
data("prec")
data("temp")

prec <- unwrap(prec)
temp <- unwrap(temp)
PAM <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Preciptation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, vect(wrld_simpl))

# Create environmental PAM
res <- lets.envpam(PAM, envs)

# Plot both spaces
lets.plot.envpam(x = res,
            species = NULL,
            cell_id_env = NULL,
            cell_id_geo = NULL,
            geo_plot = TRUE,
            world = TRUE,
            mar = c(4, 4, 4, 4))

# Highlight a single species
lets.plot.envpam(res, species = "Phyllomedusa_atlantica")

## End(Not run)
```

lets.presab — *Create a presence-absence matrix of species' geographic ranges within a grid*

Description

Convert species' ranges (in shapefile format) into a presence-absence matrix based on a user-defined grid system

Usage

```
lets.presab(
  shapes,
  xmn = NULL,
  xmx = NULL,
  ymn = NULL,
  ymx = NULL,
  resol = NULL,
  remove.cells = TRUE,
  remove.sp = TRUE,
  show.matrix = FALSE,
  crs = "+proj=longlat +datum=WGS84",
  crs.grid = crs,
  cover = 0,
  presence = NULL,
  origin = NULL,
  seasonal = NULL,
  count = FALSE
)
```

Arguments

`shapes`	Object of class `SpatVect` or `Spatial` (see packages terra and sf to read these files) containing the distribution of one or more species. Species names should be stored in the object as BINOMIAL/binomial or SCINAME/sciname.
`xmn`	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`xmx`	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`ymn`	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.

`ymx`	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`resol`	Numeric vector of length 1 or 2 to set the grid resolution. If NULL, resolution will be equivalent to 1 degree of latitude and longitude.
`remove.cells`	Logical, if `TRUE` the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).
`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`show.matrix`	Logical, if `TRUE` only the presence-absence matrix will be returned.
`crs`	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) of the polygons.
`crs.grid`	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) for the grid. Note that when you change this options you may probably change the extent coordinates and the resolution.
`cover`	Percentage of the cell covered by the shapefile that will be considered for presence (values between 0 and 1).
`presence`	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data `https://www.iucnredlist.org/resources/spatial-data-download`, see metadata).
`origin`	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
`seasonal`	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).
`count`	Logical, if `TRUE` a progress bar indicating the processing progress will be shown.

Details

This function creates the presence-absence matrix based on a raster object. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if `count` argument is set `TRUE`, a counting window will open to display the progress (i.e. the polygon/shapefile that the function is working on). Note that the number of polygons is not the same as the number of species (i.e. a species may have more than one polygon/shapefiles).

Value

The result is a list object of class `PresenceAbsence` with the following objects: **Presence-Absence Matrix**: A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used); **Richness Raster**: A raster containing species richness data; **Species name**: A character vector with species' names contained in the matrix. *But see the optional argument `show.matrix`.

Author(s)

Bruno Vilela & Fabricio Villalobos

See Also

plot.PresenceAbsence

lets.presab.birds

lets.shFilter

Examples

```
## Not run:
# Spatial distribution polygons of South American frogs
# of genus Phyllomedusa.
data(Phyllomedusa)
PAM <- lets.presab(Phyllomedusa)
summary(PAM)
# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")
# Map of individual species
plot(PAM, name = "Phyllomedusa nordestina")

## End(Not run)
```

---

lets.presab.birds — *Create a presence-absence matrix of species' geographic ranges within a grid for the Birdlife spatial data*

Description

Convert species' ranges (in shapefile format and stored in particular folders) into a presence-absence matrix based on a user-defined grid. This function is specially designed to work with BirdLife Intl. shapefiles (https://www.birdlife.org). (Notice that new versions of birds spatial data are in a similar format to other groups and should be run using the lets.presab function. We will keep this function in case someone needs to use on the previous data format.)

Usage

```
lets.presab.birds(
  path,
  xmn = NULL,
  xmx = NULL,
  ymn = NULL,
  ymx = NULL,
  resol = NULL,
  remove.cells = TRUE,
  remove.sp = TRUE,
  show.matrix = FALSE,
  crs = "+proj=longlat +datum=WGS84",
  crs.grid = crs,
  cover = 0,
  presence = NULL,
  origin = NULL,
  seasonal = NULL,
  count = FALSE
)
```

Arguments

`path`	Path location of folders with one or more species' individual shapefiles. Shapefiles with more than one species will not work on this function. To use multi-species shapefiles see `lets.presab`.
`xmn`	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`xmx`	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`ymn`	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`ymx`	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`resol`	Numeric vector of length 1 or 2 to set the grid resolution. If NULL, resolution will be equivalent to 1 degree of latitude and longitude.
`remove.cells`	Logical, if `TRUE` the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).
`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`show.matrix`	Logical, if `TRUE` only the presence-absence matrix will be returned.
`crs`	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) of the polygons.
`crs.grid`	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) for the grid. Note that when you change this options you may probably change the extent coordinates and the resolution.
`cover`	Percentage of the cell covered by the shapefile that will be considered for presence (values between 0 and 1).
`presence`	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data `https://www.iucnredlist.org/resources/spatial-data-download`, see metadata).
`origin`	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).

`seasonal`	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).
`count`	Logical, if `TRUE` a progress bar indicating the processing progress will be shown.

Details

The function creates the presence-absence matrix based on a raster file. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if `count` argument is set `TRUE`, a counting window will open so you can see the progress (i.e. in what polygon the function is working). Note that the number of polygons is not the same as the number of species that you have (i.e. a species may have more than one polygon/shapefiles).

Value

The result is a list object of class `PresenceAbsence` with the following objects: **Presence-Absence Matrix**: A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used); **Richness Raster**: A raster containing species richness data; **Species name**: A character vector with species' names contained in the matrix. *But see the optional argument `show.matrix`.

Author(s)

Bruno Vilela & Fabricio Villalobos

See Also

`plot.PresenceAbsence`

`lets.presab`

`lets.shFilter`

Examples

```
## Not run:
# Constructing a Presence/Absence matrix for birds
# Attention: For your own files, omit the 'system.file'
# and 'package="letsR"', these are just to get the path
# to files installed with the package.
path.Ramphastos <- system.file("extdata", package = "letsR")
PAM <- lets.presab.birds(path.Ramphastos, xmn = -93, xmx = -29,
                         ymn = -57, ymx = 25)

# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Ramphastos species Richness")


## End(Not run)
```

`lets.presab.grid` *Create a presence-absence matrix of species' geographic ranges within a user's grid shapefile (beta version)*

Description

Convert species' ranges (in shapefile format) into a presence-absence matrix based on a grid in shapefile format.

Usage

```
lets.presab.grid(
  shapes,
  grid,
  sample.unit,
  remove.sp = TRUE,
  presence = NULL,
  origin = NULL,
  seasonal = NULL
)
```

Arguments

`shapes`	Object of class `SpatVect` or `Spatial` (see packages terra and sf to read these files) containing the distribution of one or more species. Species names should be stored in the object as BINOMIAL/binomial or SCINAME/sciname.
`grid`	Object of class SpatVector (see function `terra::vect`) representing the spatial grid (e.g. regular/irregular cells, political divisions, hexagonal grids, etc). The grid and the shapefiles must be in the same projection.
`sample.unit`	Object of class `character` with the name of the column (in the grid) representing the sample units of the presence absence matrix.
`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`presence`	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data `https://www.iucnredlist.org/resources/spatial-data-download`, see metadata).
`origin`	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
`seasonal`	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).

Details

This function is an alternative way to create a presence absence matrix when users already have their own grid.

Value

The result is a `list` containing two objects:

(I) A matrix the species presence (1) and absence (0) values per sample unity.

(II) The original grid.

Author(s)

Bruno Vilela & Fabricio Villalobos

See Also

`lets.presab`

`lets.presab.grid.points`

`lets.shFilter`

Examples

```
## Not run:
# Species polygons
data("Phyllomedusa")
data("wrld_simpl")
# Grid
sp.r <- terra::as.polygons(terra::rast(resol = 5,
crs = terra::crs(Phyllomedusa),
xmin = -93, xmax = -29, ymin = -57, ymax = 15))
sp.r$ID <- 1:length(sp.r)


# PAM
resu <- lets.presab.grid(Phyllomedusa, sp.r, "ID")

# Plot
rich_plus1 <- rowSums(resu$PAM[, -1]) + 1
colfunc <- colorRampPalette(c("#fff5f0", "#fb6a4a", "#67000d"))
colors <- c("white", colfunc(max(rich_plus1)))
plot(resu$grid, border = "gray40",
     col = colors[rich_plus1])
plot(sf::st_geometry(wrld_simpl), add = TRUE)

## End(Not run)
```

`lets.presab.grid.points`

Create a presence-absence matrix based on species' point occurrences within a user's grid shapefile (beta version)

Description

Convert species' occurrences points into a presence-absence matrix based on a grid in shapefile format.

Usage

```
lets.presab.grid.points(
  xy,
  species,
  grid,
  sample.unit,
  remove.sp = TRUE,
  abundance = TRUE
)
```

Arguments

`xy`	A matrix with geographic coordinates of species occurrences, first column is the longitude (or x), and the second latitude (or y).
`species`	Character vector with species names, in the same order as the coordinates.
`grid`	Object of class SpatVector (see function `terra::vect`) representing the spatial grid (e.g. regular/irregular cells, political divisions, hexagonal grids, etc). The grid and the shapefiles must be in the same projection.
`sample.unit`	Object of class `character` with the name of the column (in the grid) representing the sample units of the presence absence matrix.
`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`abundance`	Logical. If TRUE, the resulting matrix will display number of occurrences per species in each cell. If FALSE, the resulting matrix will show presence-absence values.

Details

This function is an alternative way to create a presence absence matrix when users already have their own grid.

Value

The result is a `list` containing two objects:

(I) A matrix the species presence (1) and absence (0) values per sample unity.

(II) The original grid.

Author(s)

Bruno Vilela & Fabricio Villalobos

See Also

lets.presab.points

lets.presab.grid

Examples

```
## Not run:
# Species polygons
data("wrld_simpl")

# Grid
crs = "+proj=longlat +datum=WGS84 +no_defs"
sp.r <- terra::as.polygons(terra::rast(
  resol = 5,
  crs = crs,
  xmin = -93,
  xmax = -29,
  ymin = -57,
  ymax = 15
))
sp.r$ID <- 1:length(sp.r)

# Occurrence Points
N = 20
species <- c(rep("sp1", N), rep("sp2", N),
             rep("sp3", N), rep("sp4", N))
x <- runif(N * 4, min = -69, max = -51)
y <- runif(N * 4, min = -23, max = -4)
xy <- cbind(x, y)

# PAM
resu <- lets.presab.grid.points(xy, species, sp.r, "ID", abundance = FALSE)

# Plot
rich_plus1 <- rowSums(resu$PAM[, -1, drop = FALSE]) + 1
colfunc <- colorRampPalette(c("#fff5f0", "#fb6a4a", "#67000d"))
colors <- c("white", colfunc(max(rich_plus1)))
occs <- terra::vect(xy, crs = crs)
plot(resu$grid, border = "gray40",
     col = colors[rich_plus1])
plot(sf::st_geometry(wrld_simpl), add = TRUE)
plot(occs, cex = 0.5, col = rep(1:4, each = N), add = T)

## End(Not run)
```

lets.presab.points *Create a presence-absence matrix based on species' point occurrences*

Description

Convert species' occurrences points into a presence-absence matrix based on a user-defined grid.

Usage

```
lets.presab.points(
  xy,
  species,
  xmn = NULL,
  xmx = NULL,
  ymn = NULL,
  ymx = NULL,
  resol = NULL,
  remove.cells = TRUE,
  remove.sp = TRUE,
  show.matrix = FALSE,
  crs = "+proj=longlat +datum=WGS84",
  count = FALSE
)
```

Arguments

`xy`	A matrix with geographic coordinates of species occurrences, first column is the longitude (or x), and the second latitude (or y).
`species`	Character vector with species names, in the same order as the coordinates.
`xmn`	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`xmx`	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`ymn`	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`ymx`	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest). If NULL, limits will be calculated based on the limits of the shapes object.
`resol`	Numeric vector of length 1 or 2 to set the grid resolution. If NULL, resolution will be equivalent to 1 degree of latitude and longitude.
`remove.cells`	Logical, if `TRUE` the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).

`remove.sp`	Logical, if `TRUE` the final matrix will not contain species that do not match any cell in the grid.
`show.matrix`	Logical, if `TRUE` only the presence-absence matrix will be returned.
`crs`	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) of the polygons.
`count`	Logical, if `TRUE` a progress bar indicating the processing progress will be shown.

Details

The function creates the presence-absence matrix based on a raster file. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if `count` argument is set `TRUE`, a counting window will open so you can see the progress (i.e. in what polygon the function is working). Note that the number of polygons is not the same as the number of species that you have (i.e. a species may have more than one polygon/shapefiles).

Value

The result is a list object of class `PresenceAbsence` with the following objects: **Presence-Absence Matrix**: A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used); **Richness Raster**: A raster containing species richness data; **Species name**: A character vector with species' names contained in the matrix. *But see the optional argument `show.matrix`.

Author(s)

Bruno Vilela & Fabricio Villalobos

See Also

`plot.PresenceAbsence`

`lets.presab.birds`

`lets.presab`

`lets.shFilter`

Examples

```
## Not run:
species <- c(rep("sp1", 100), rep("sp2", 100),
             rep("sp3", 100), rep("sp4", 100))
x <- runif(400, min = -69, max = -51)
y <- runif(400, min = -23, max = -4)
xy <- cbind(x, y)
PAM <- lets.presab.points(xy, species, xmn = -93, xmx = -29,
                          ymn = -57, ymx = 15)
summary(PAM)
# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Species richness map (simulated)")
```

```
# Map of the specific species
plot(PAM, name = "sp1")

## End(Not run)
```

lets.rangesize *Compute species' geographic range sizes*

Description

This function calculates species' range sizes from a PresenceAbsence object or directly from the species' shapefiles.

Usage

```
lets.rangesize(
  x,
  species_name = NULL,
  coordinates = "geographic",
  units = "cell"
)
```

Arguments

`x`	A `PresenceAbsence` or an `SpatVector` object.
`species_name`	Species names in the same order as in the `SpatVector` (only needed if x is a `SpatVector`).
`coordinates`	"geographical" or "planar". Indicate whether the shapefile has geographical or planar coordinates(only needed if x is a `SpatVector`).
`units`	"cell" or "squaremeter". Indicate if the size units wanted are in number of cells occupied or in square meters(only needed if x is a `PresenceAbsence` object).

Value

The result is a matrix with the range size of each species. If the range size accounts for the earth curvature (Yes or No) or its size unit may differ for each argument combination:

1) SpatVector & geographical = Square meters. Yes.

2) SpatVector & planar = Square meters. No.

3) PresenceAbsence & cell = number of cells. No.

4) PresenceAbsence & squaremeter = Square meters. Yes.

Author(s)

Bruno Vilela

Examples

```
## Not run:
# SpatialPolygonsDataFrame & geographical
data(Phyllomedusa)
rangesize <- lets.rangesize(x = Phyllomedusa,
                            coordinates = "geographic")

# SpatialPolygonsDataFrame & planar
rangesize2 <- lets.rangesize(x = Phyllomedusa,
                             coordinates = "planar")

# PresenceAbsence & cell
data(PAM)
rangesize3 <- lets.rangesize(x = PAM,
                             units = "cell")

# PresenceAbsence & squaremeter
rangesize4 <- lets.rangesize(x = PAM,
                             units = "squaremeter")

## End(Not run)
```

`lets.save` *Save a PresenceAbsence object*

Description

Save an external representation of a PresenceAbsence object to the specified file. The object can be read back from the file at a later date by using the function `lets.load`.

Usage

```
lets.save(pam, ...)
```

Arguments

`pam`	A PresenceAbsence object.
`...`	other arguments passed to the function `save`

Author(s)

Bruno Vilela

See Also

`lets.presab`

`lets.presab.birds`

Examples

```
## Not run:
data(PAM)
lets.save(PAM, file = "PAM.RData")
PAM <- lets.load(file = "PAM.RData")

## End(Not run)
```

`lets.shFilter` *Filter species' shapefiles based on its presence, origin, and season*

Description

Filter species shapefiles by origin, presence, and seasonal type (following IUCN types: `https://www.iucnredlist.org/resources/spatial-data-download`, see metadata).

Usage

```
lets.shFilter(shapes, presence = NULL, origin = NULL, seasonal = NULL)
```

Arguments

`shapes`	Object of class `SpatVect` or `Spatial` (see packages terra and sf to read these files) containing the distribution of one or more species. Species names should be stored in the object as BINOMIAL/binomial or SCINAME/sciname.
`presence`	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data `https://www.iucnredlist.org/resources/spatial-data-download`, see metadata).
`origin`	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
`seasonal`	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).

Details

Presence codes: (1) Extant, (2) Probably Extant, (3) Possibly Extant, (4) Possibly Extinct, (5) Extinct (post 1500) & (6) Presence Uncertain.

Origin codes: (1) Native, (2) Reintroduced, (3) Introduced, (4) Vagrant & (5) Origin Uncertain.

Seasonal codes: (1) Resident, (2) Breeding Season, (3) Non-breeding Season, (4) Passage & (5) Seasonal Occurrence Uncertain.

More info in the shapefiles' metadata.

Value

The result is the shapefile(s) filtered according to the selected types. If the filters remove all polygons, the result will be NULL.

Author(s)

Bruno Vilela

See Also

plot.PresenceAbsence

lets.presab

lets.presab.birds

Examples

```
## Not run:
data(Phyllomedusa)

filtered_shape <- lets.shFilter(
  shape = Phyllomedusa,
  presence = 1,
  origin = 1,
  seasonal = 1)

if (!is.null(filtered_shape)) {
   plot(filtered_shape, col = "lightgreen", border = "darkgreen")
}

## End(Not run)
```

---

lets.subsetPAM *Subset a PresenceAbsence object based on species names*

Description

Subset a PresenceAbsence object based on species character vector provided by the user.

Usage

```
lets.subsetPAM(x, names, remove.cells = TRUE)
```

Arguments

x	A PresenceAbsence object.
names	Character vector with species names to subset the PresenceAbsence object.
remove.cells	Logical, if TRUE the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).

Value

The result is an object of class PresenceAbsence subseted.

Author(s)

Bruno Vilela

See Also

plot.PresenceAbsence

lets.presab.birds

Examples

```
## Not run:
data(PAM)
# PAM before subset
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")

# Subset PAM to the first 20 species
PAMsub <- lets.subsetPAM(PAM, PAM[[3]][1:20])
plot(PAMsub, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")

## End(Not run)
```

---

lets.summarizer — *Summarize variable(s) values in a presence-absence matrix within species' ranges*

Description

Based on a Presence-Absence matrix with added variables (see `lets.addvar`), this function summarizes the values of such variable(s) per species (across the species' occupied cells. i.e. within their ranges).

Usage

```
lets.summarizer(x, pos, xy = TRUE, fun = mean, ...)
```

Arguments

`x`	Presence-absence matrix with variables added.
`pos`	Column position of the variables of interest.
`xy`	Logical, if `TRUE` the input matrix contains geographic coordinates in the first two columns.
`fun`	Function to be used to summarize the variable per species. Default is `mean`.
`...`	Other parameters passed to the function defined in `fun`.

Author(s)

Bruno Vilela & Fabricio Villalobos

References

Villalobos, F. and Arita, H.T. 2010. The diversity field of New World leaf-nosed bats (Phyllostomidae). Global Ecology and Biogeography. 19, 200-211.

See Also

lets.addvar

lets.addpoly

lets.field

Examples

```
## Not run:
data(PAM)
data(temp)
temp <- terra::unwrap(temp)
pamvar <- lets.addvar(PAM, temp)
resu <- lets.summarizer(x = pamvar, pos = ncol(pamvar),
                        xy = TRUE)

## End(Not run)
```

`lets.summaryze.cells` *Aggregate cell-wise metrics across occupied cells per species*

Description

Given a presence–absence matrix in either attribute space (from `lets.attrpam`) or environmental space (from `lets.envpam`) and a table of per-cell descriptors (from `lets.attrcells` or `lets.envcells`), this function aggregates each descriptor across the set of cells occupied by each species, using a user-supplied summary function (e.g., `mean`, `median`, `sum`).

Usage

```
lets.summaryze.cells(x, y, func = mean)
```

Arguments

`x`	A list returned by `lets.attrpam` (attribute space) or `lets.envpam` (environmental space). The first element `x[[1]]` must be a matrix/data.frame whose first three columns are the cell identifier and the two coordinate/axis columns; species presence–absence columns start at column 4.

`y`	A `data.frame` of per-cell descriptors with the same row order as `x[[1]]`. The first column must be the cell identifier (e.g., `Cell_attr` or `Cell_env`); descriptor columns start at column 2.
`func`	A summary function to aggregate a descriptor across the cells occupied by a species (default `mean`). The function must return a single numeric value and should handle `na.rm` internally if needed (note that this routine already removes `NA`s per species via `na.omit`).

Details

Summarize per-cell descriptors to species-level values

Internally, for each species column in `x[[1]]` (starting at column 4), the function builds a logical mask of occupied cells (> 0). It then subsets the descriptor table `y` to those rows and applies `func` column-wise to `y[occupied, -1]` (dropping the ID column). Missing values are removed with `stats::na.omit` prior to aggregation.

Value

A `data.frame` with one row per species and one column per descriptor, containing the aggregated (e.g., mean) value of each descriptor across all cells where that species is present. The first column, `Species`, holds the species names copied from `colnames(x[[1]])[4:ncol(x[[1]])]`.

See Also

`lets.attrpam`, `lets.envpam`, `lets.attrcells`, `lets.envcells`

Examples

```
## Not run:
## --------------------------
## ATTRIBUTE SPACE WORKFLOW
## --------------------------
set.seed(1)
n <- 2000
Species <- paste0("sp", 1:n)
trait_a <- rnorm(n)
trait_b <- trait_a * 0.2 + rnorm(n)
df <- data.frame(Species, trait_a, trait_b)

# Build AttrPAM and compute per-cell descriptors
x_attr <- lets.attrpam(df, n_bins = 30)
y_attr <- lets.attrcells(x_attr)

# Summarize descriptors per species (e.g., mean across occupied cells)
head(lets.summaryze.cells(x_attr, y_attr, func = mean))

## ------------------------------
## ENVIRONMENTAL SPACE WORKFLOW
## ------------------------------
data("Phyllomedusa"); data("prec"); data("temp")
prec <- unwrap(prec); temp <- unwrap(temp)
```

```
PAM  <- lets.presab(Phyllomedusa, remove.cells = FALSE)
envs <- lets.addvar(PAM, c(temp, prec), onlyvar = TRUE)
colnames(envs) <- c("Temperature", "Precipitation")
wrld_simpl <- get(utils::data("wrld_simpl", package = "letsR"))
PAM <- lets.pamcrop(PAM, terra::vect(wrld_simpl))

res_env <- lets.envpam(PAM, envs, n_bins = 30)
y_env   <- lets.envcells(res_env, perc = 0.2)

# Summarize environmental-space descriptors per species
head(lets.summaryze.cells(res_env, y_env, func = mean))

## End(Not run)
```

`lets.transf` *Transform values of a vector*

Description

Transform each element of a vector.

Usage

```
lets.transf(x, y, z, NUMERIC = TRUE)
```

Arguments

`x`	A vector to be transformed.
`y`	levels to be transformed.
`z`	The value to be atributed to each level (same order as y).
`NUMERIC`	logical, if `TRUE` z will be considered numbers.

Value

Return a vector with changed values.

Author(s)

Bruno Vilela

Examples

```
## Not run:
status <- sample(c("EN","VU", "NT", "CR", "DD", "LC"), 30, replace=TRUE)
TE <- "Threatened"
NT <- "Non-Threatened"
new <- c(TE, TE, NT, TE, "Data Deficient", NT)
```

```
old <- c("EN","VU", "NT", "CR", "DD", "LC")
statustrans <- lets.transf(status, old, new, NUMERIC=FALSE)


## End(Not run)
```

`PAM` *PresenceAbsence object for frogs of the genus Phyllomedusa*

Description

PresenceAbsence object obtained using the function `lets.presab` for the Geographic distribution of the South American frog genus Phyllomedusa.

Usage

```
PAM
```

Format

A PresenceAbsence object

Source

Generated from IUCN Spatial Data - `https://www.iucnredlist.org/`. 2014.

`Phyllomedusa` *Phyllomedusa*

Description

Geographic distribution of the South American frog genus Phyllomedusa.

Usage

```
Phyllomedusa
```

Format

A simple feature collection for 32 species with 46 features and 4 fields.

binomial Scientific name

presence IUCN Red List distributional code

origin IUCN Red List distributional code

seasonal IUCN Red List distributional code

Details

See the IUCN Red List Attribute Information for Species Ranges.

Source

IUCN - `https://www.iucnredlist.org/`. 2014.

`plot.PresenceAbsence` *Plot an object of class PresenceAbsence*

Description

Plots species richness map from an object of class PresenceAbsence or a particular species' map.

Usage

```
## S3 method for class 'PresenceAbsence'
plot(x, name = NULL, world = TRUE, col_rich = NULL, col_name = "red", ...)
```

Arguments

`x`	An object of class `PresenceAbsence`.
`name`	A character specifying a species to be ploted instead of the complete species richness map.
`world`	If `TURE` a map of political divisions (countries) is added to the plot.
`col_rich`	Color function (e.g. `rainbow`, `heat.colors`, `colorRampPalette`) to be used in the richness map.
`col_name`	The color to use when ploting single species.
`...`	Other parameters pass to the plot function.

Author(s)

Bruno Vilela

See Also

`lets.presab`

`lets.presab.birds`

Examples

```
## Not run:
data(PAM)
plot(PAM)
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")
plot(PAM, name = "Phyllomedusa atelopoides")
plot(PAM, name = "Phyllomedusa azurea")

## End(Not run)
```

---

prec *Annual Precipitation raster for the world.*

Description

Annual Precipitation in mm for the world in 10 arc min of resolution.

Usage

```
prec
```

Format

A PackedStatRaster object.

Source

Data was modified from WorldClim (https://worldclim.org/, downloaded 10/2024).

Hijmans, R.J., S.E. Cameron, J.L. Parra, P.G. Jones and A. Jarvis, 2005. Very high resolution interpolated climate surfaces for global land areas. International Journal of Climatology 25: 1965-1978.

PresenceAbsence *PresenceAbsence Class*

Description

The `PresenceAbsence` is a new S3 object class created and used inside the `letsR` package. This object class is used to store information on species distribution within a geographic domain in the form of a presence-absence matrix. In addition, the `PresenceAbsence` object also contains other essential information (e.g. user-defined grid cell system, including resolution, projection, datum, and extent) necessary for other analysis performed with the package's functions.

Details

Creating a PresenceAbsence object
A `PresenceAbsence` object can be generated using the following functions:
- `lets.presab`
- `lets.presab.birds`
- `lets.presab.points`

The PresenceAbsence information
The result is a `list` object of class `PresenceAbsence` that includes the following objects:
- Presence_and_Absence_Matrix: A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used);
- Richness_Raster: A raster containing species richness information across the geographic domain, which can be used to map the observed geographic gradient in species richness;
- Species_name: A character vector with species' names contained in the matrix.

Each of the objects can be obtained usign the standard subsetting operators that are commonly applied to a `list` object (i.e. '[[' and '$').

letsR functions applied to a PresenceAbsence object
The following functions from the letsR package can be directly applied to a `PresenceAbsence`:
- `lets.addpoly`
- `lets.addvar`
- `lets.distmat`
- `lets.field`
- `lets.gridirizer`
- `lets.maplizer`
- `lets.midpoint`
- `lets.overlap`
- `lets.pamcrop`
- `lets.rangesize`

Generic functions applied to a PresenceAbsence object
The following generic functions can be directly applied to the `PresenceAbsence` object.
- `print` (`print.PresenceAbsence`)
- `summary` (`summary.PresenceAbsence`)
- `plot` (`plot.PresenceAbsence`)

`print.PresenceAbsence` *Print for object of class PresenceAbsence*

Description

Print for objects of class PresenceAbsence.

Usage

```
## S3 method for class 'PresenceAbsence'
print(x, ...)
```

Arguments

x	an object of class PresenceAbsence.
...	Other print parameters.

Author(s)

Bruno Vilela

`print.summary.PresenceAbsence`
Print summary for object of class PresenceAbsence

Description

Print summary for objects of class PresenceAbsence.

Usage

```
## S3 method for class 'PresenceAbsence'
print.summary(x, ...)
```

Arguments

x	an object of class PresenceAbsence.
...	Other print parameters.

Author(s)

Bruno Vilela

summary.PresenceAbsence

Summary for object of class PresenceAbsence

Description

Summary for objects of class PresenceAbsence.

Usage

```
## S3 method for class 'PresenceAbsence'
summary(object, ...)
```

Arguments

`object`	an object of class `PresenceAbsence`.
`...`	additional arguments affecting the summary produced.

Author(s)

Bruno Vilela

temp

Mean temperature raster for the world.

Description

Mean temperature raster in degrees Celsius (multiplied by 100) for the world in 10 arc min of resolution.

Usage

```
temp
```

Format

A PackedStatRaster object.

Source

Data was modified from WorldClim (`https://worldclim.org/`, downloaded 05/2014).

Hijmans, R.J., S.E. Cameron, J.L. Parra, P.G. Jones and A. Jarvis, 2005. Very high resolution interpolated climate surfaces for global land areas. International Journal of Climatology 25: 1965-1978.

`wrld_simpl` *Simplified world country polygons*

Description

World map in sf format. Obtained from maptools and converted to sf.

Usage

```
wrld_simpl
```

Format

A simple feature collection with 246 features and 11 fields.

Source

Now available from `https://github.com/nasa/World-Wind-Java/tree/master/WorldWind/testData/shapefiles`.

Index