Package ‘USE’

September 15, 2025

Title Uniform Sampling of the Environmental Space

Version 0.1.6

Description Provides functions for uniform sampling of the environmental space, designed to assist species distribution modellers in gathering ecologically relevant pseudo-absence data. The method ensures balanced representation of environmental conditions and helps reduce sampling bias in model calibration. Based on the framework described by Da Re et al. (2023) <doi:10.1111/2041-210X.14209>.

Depends R (>= 3.6.0)

Imports sf, parallel, terra, ks, ggplot2, cowplot

License GPL (>= 2)

Encoding UTF-8

LazyData true

URL `https://danddr.github.io/USE/`, `https://github.com/danddr/USE`

BugReports `https://github.com/danddr/USE/issues`

RoxygenNote 7.3.2

Suggests rmarkdown, knitr, tidyterra

VignetteBuilder knitr

NeedsCompilation no

Author Daniele Da Re [aut, cre] (ORCID: <`https://orcid.org/0000-0002-3398-9295`>), Enrico Tordoni [aut] (ORCID: <`https://orcid.org/0000-0002-9722-6692`>), Manuele Bazzichetto [aut] (ORCID: <`https://orcid.org/0000-0002-9874-5064`>)

Maintainer Daniele Da Re <`dare.daniele@gmail.com`>

Repository CRAN

Date/Publication 2025-09-15 08:10:02 UTC

Contents

`optimRes` *Get optimal resolution of the sampling grid*

Description

`optimRes` identifies the optimal resolution of the sampling grid to be used to perform the uniform environmental sampling. To find this optimal resolution, a set of candidate resolutions must be provided. For each candidate resolution, `optimRes` calculates a metric that summarizes the average squared Euclidean distance between the observations (PC-scores of the first two principal components) within each cell and the centroid of the convex hull encompassing the points. It's important to note that the centroid is specific to each cell.

Usage

```
optimRes(sdf, grid.res, perc.thr = 10, cr = 1, showOpt = TRUE)
```

Arguments

`sdf`	an sf object having point geometry given by the PC-scores values
`grid.res`	(integer) a vector of resolutions to be tested, i.e seq(1,100, by=1)
`perc.thr`	rate of change (expressed in percentage) of the function to be minimized for selecting the optimal resolution.
`cr`	(integer) number of cores for parallel computing. The default cluster type is PSOCK.
`showOpt`	(logical) plot the result.

Details

This metric is then compared across different sampling grids with increasing resolution, i.e., an increasing number of cells. The best resolution is selected based on the trade-off between the number of cells and the average distance among observations within each cell. Essentially, the goal is to find the finest resolution of the sampling grid that enables uniform sampling of the environmental space without overfitting it.

By default, the optimal resolution is determined as the one where the average distance among observations and the cell-specific centroids cannot be reduced by more than 10%. However, users have the flexibility to adjust this setting according to their needs. The `optimRes` function returns a list with two elements. The first element is a matrix that reports the metric calculated for each sampling grid at the corresponding resolution. The second element is the selected optimal resolution.

Additionally, the function provides a plot that displays the metric values for each resolution. This allows users to visually analyze the relationship between resolution and the associated metric, thereby empowering them to make an informed decision when selecting a resolution.

In case the function returns NA as the optimal resolution: i) increase the range of `grid.res`, ii) increase `perc.thr`.

Value

It returns a list with: i) a matrix reporting the values of the function to be minimized, along with the corresponding resolution; ii) the optimal resolution.

`paSampling` *Sampling pseudo-absences for the training and testing datasets.*

Description

`paSampling` performs a two-step procedure for uniformly sampling pseudo-absences within the environmental space. In the initial step, a kernel-based filter is utilized to determine the subset of the environmental space that will be subsequently sampled. The kernel-based filter calculates the probability function based on the presence observations, enabling the identification of areas within the environmental space that likely exhibit suitable conditions for the species. To achieve this, a probability threshold value is utilized to assign observations to the corresponding portion of the environmental space. These areas, deemed to have suitable environmental conditions, are excluded from the subsequent uniform sampling process conducted in the second step using the `uniformSampling` function, which is internally called. The bandwidth of the kernel can be automatically estimated from the presence observations or directly set by the user, providing flexibility in determining the scope and precision of the filter.

Usage

```
paSampling(
  env.rast = NULL,
  pres = NULL,
  thres = 0.75,
  H = NULL,
  grid.res = NULL,
  n.tr = 5,
  sub.ts = FALSE,
  n.ts = 5,
  prev = NULL,
  plot_proc = FALSE,
  verbose = FALSE
)
```

Arguments

`env.rast`	A RasterStack, RasterBrick or a SpatRaster object comprising the variables describing the environmental space.
`pres`	A SpatialPointsDataframe, a SpatVector or an sf object including the presence-only observations of the species of interest.
`thres`	(double) This value identifies the quantile value used to specify the boundary of the kernel density estimate (default `thres=0.75`). Thus, probability values higher than the threshold should indicate portions of the multivariate space likely associated with presence points.
`H`	The kernel bandwidth (i.e., the width of the kernel density function that defines its shape) excluding the portion of the environmental space associated with environmental conditions likely suitable for the species. It can be either defined by the user or automatically estimated by `paSampling` via `ks::Hpi`.
`grid.res`	(integer) resolution of the sampling grid. The resolution can be arbitrarily selected or defined using the `optimRes` function.
`n.tr`	(integer) number of pseudo-absences for the training dataset to sample in each cell of the sampling grid
`sub.ts`	(logical) sample the validation pseudo-absences
`n.ts`	(integer; optional) number of pseudo-absences for the testing dataset to sample in each cell of the sampling grid. sub.ts argument must be TRUE.
`prev`	(double) prevalence value to be specified instead of n.tr and n.ts
`plot_proc`	(logical) plot progress of the sampling, default FALSE
`verbose`	(logical) Print verbose

Details

Being designed with species distribution models in mind, `paSampling` allows collectively sampling pseudo-absences for both the training and testing dataset (optional). In both cases, the user must provide a number of observations that will be sampled in each cell of the sampling grid (`n.tr`: points for the training dataset; `n.ts`: points for the testing dataset). Note that the optimal resolution of the sampling grid can be found using the `optimRes` function. Also, note that the number of pseudo-absences eventually sampled in each cell by the internally-called `uniformSampling` function depends on the spatial configuration of the observations within the environmental space. Indeed, in most cases some cells of the sampling grid will be empty (i.e., those at the boundary of the environmental space). For this reason, the number of pseudo-absences returned by `paSampling` is likely to be lower than the product between the number of cells of the sampling gird and `n.tr`(or `n.ts`).

Value

An sf object with the coordinates of the pseudo-absences both in the geographical and environmental space.

`rastPCA` *Principal Component Analysis for Rasters*

Description

The `rastPCA` function calculates the principal component analysis (PCA) for SpatRaster, RasterBrick, or RasterStack objects and returns a SpatRaster with multiple layers representing the PCA components. Internally, `rastPCA` utilizes the princomp function for R-mode PCA analysis. The covariance matrix is computed using all the observations within the provided SpatRaster object, which describes the environmental conditions. The covariance matrix obtained is subsequently utilized as input for the `princomp` function, which conducts the PCA. The resulting PCA components are then used to generate the final SpatRaster, consisting of multiple layers that represent the PCA components.

Usage

```
rastPCA(env.rast, nPC = NULL, naMask = TRUE, stand = FALSE)
```

Arguments

`env.rast`	A RasterStack, RasterBrick or a SpatRaster object comprising the variables describing the environmental space.
`nPC`	Integer. Number of PCA components to return.
`naMask`	Logical. Masks all pixels which have at least one NA (default `TRUE` is recommended but introduces a slow-down.
`stand`	Logical. If `TRUE`, perform standardized PCA. Corresponds to centered and scaled input image. This is usually beneficial for equal weighting of all layers. (`FALSE` by default)

Details

Pixels with missing values in one or more bands will be set to NA. The built-in check for such pixels can lead to a slow-down of rastPCA. However, if you make sure or know beforehand that all pixels have either only valid values or only NAs throughout all layers you can disable this check by setting `naMask=FALSE` which speeds up the computation.

Standardized PCA (`stand=TRUE`) can be useful if imagery or bands of different dynamic ranges are combined. In this case, the correlation matrix is computed instead of the covariance matrix, which has the same effect as using normalised bands of unit variance.

Value

Returns a named list containing the PCA model object ($pca) and the SpatRaster with the principal component layers ($PCs).

See Also

The `rastPCA` function has been conceptualized starting from `RStoolbox::rasterPCA` (`https://github.com/bleutner/RStoolbox`).

`SpatialProba` *Virtual species probability of occurrence*

Description

The `SpatialProba` function calculates the simulated probability of occurrence of a virtual species based on an additive model that incorporates environmental variables. The model considers both linear and quadratic relationships between the environmental factors and the species' probability of presence. This function uses environmental data provided as a SpatRaster object (e.g., temperature, precipitation) to compute the probability of species presence across a defined area of interest. The resulting probabilities are mapped to a range between 0 and 1, representing the likelihood of species occurrence in the given locations.

Usage

```
SpatialProba(coefs, env.rast, quadr_term, marginalPlots)
```

Arguments

`coefs`	a named vector of regression parameters. Names must match those of the environmental layers (except for intercept, and quadratic terms). Parameters for quadratic terms must have the prefix 'quadr_' (e.g., `quadr_bio1`).
`env.rast`	a SpatRaster object with environmental layers to generate the spatial layer of probabilities.
`quadr_term`	a named vector with names of coefs for which a quadratic term is specified (without prefix 'quadr_').
`marginalPlots`	logical, if TRUE, returns marginal plots.

Value

A list containing a SpatRaster with the species' occurrence probability and, if `marginalPlots=TRUE`, a graphical plot of the response curves.

`thresh.inspect` *Inspect the effect of the kernel threshold parameter on the environmental space partitioning*

Description

`thresh.inspect` function allows for a pre-inspection of the impact that selecting a specific threshold for the kernel-based filter will have on the exclusion of the environmental space in the subsequent uniform sampling of the pseudo-absences process (see `paSampling`). By providing a range of threshold values, the function generates a plot that illustrates the entire environmental space, including the portion delineated by the kernel-based filter and the associated convex-hull. This plot helps visualize the areas that will be excluded from the uniform sampling of the pseudo-absences. This functionality proves particularly valuable in determining a meaningful threshold for the kernel-based filter in specific ecological scenarios. For instance, when dealing with sink populations, selecting the appropriate threshold enables the exclusion of environmental space regions where the species is present, but the conditions are unsuitable. This allows for a more accurate sampling of pseudo-absences, considering the unique requirements of different ecological contexts.

Usage

```
thresh.inspect(env.rast, pres = NULL, thres = 0.75, H = NULL)
```

Arguments

`env.rast`	A RasterStack, RasterBrick or a SpatRaster object comprising the variables describing the environmental space.
`pres`	A SpatialPointsDataframe, a SpatVector or an sf object including the presence-only observations of the species of interest.
`thres`	(double) This value or vector of values identifies the quantile value used to specify the boundary of the kernel density estimate (default `thres=0.75`). Thus, probability values higher than the threshold should indicate portions of the multivariate space likely associated with presence points.
`H`	The kernel bandwidth (i.e., the width of the kernel density function that defines its shape) excluding the portion of the environmental space associated with environmental conditions likely suitable for the species. It can be either defined by the user or automatically estimated by `paSampling` via `ks::Hpi`.

Value

A ggplot2 object showing how the environmental space is partitioned accordingly to the selected `thres` values.

`uniformSampling` *Uniform sampling of the environmental space*

Description

`uniformSampling` performs the uniform sampling of observations within the environmental space. Note that `uniformSampling` can be more generally used to sample observations (not necessarily associated with species occurrence data) within bi-dimensional spaces (e.g., vegetation plots). Being designed with species distribution models in mind, `uniformSampling` allows collectively sampling observations for both the training and testing dataset (optional). In both cases, the user must provide a number of observations that will be sampled in each cell of the sampling grid (`n.tr`: points for the training dataset; `n.ts`: points for the testing dataset). Note that the optimal resolution of the sampling grid can be found using the `optimRes` function.

Usage

```
uniformSampling(
  sdf,
  grid.res,
  n.tr = 5,
  n.prev = NULL,
  sub.ts = FALSE,
  n.ts = 5,
  plot_proc = FALSE,
  verbose = FALSE
)
```

Arguments

`sdf`	an sf object having point geometry given by the PC-scores values
`grid.res`	(integer) resolution of the sampling grid. The resolution can be arbitrarily selected or defined using the `optimRes()` function.
`n.tr`	(integer; optional) number of expected points given a certain prevalence threshold for the training dataset.
`n.prev`	(double) sample prevalence
`sub.ts`	(logical) sample the validation points
`n.ts`	(integer; optional) number of points for the testing dataset to sample in each cell of the sampling grid. sub.ts argument must be TRUE.
`plot_proc`	(logical) plot progress of the sampling
`verbose`	(logical) Print verbose

Value

An sf object with the coordinates of the sampled points both in the geographical and environmental space

---

`Worldclim_tmp` *A subset of WorldClim bioclimatic variables*

Description

A subset of WorldClim bioclimatic variables cropped on the Central and Western Europe.

Usage

```
data(Worldclim_tmp)
```

Format

A data frame obtained from a SpatRaster with 1080 rows, 2160 columns, and 6 layers, namely:"bio1" "bio3" "bio9" "bio12" "bio13" "bio15"

Source

`geodata::worldclim_global(var='bio', res=10, path=getwd())[[c(1, 3,9, 12, 13, 15)]]`

Index