Package 'dynamicSDM'

October 18, 2023

Title Species Distribution and Abundance Modelling at High Spatio-Temporal Resolution

Version 1.3.3

Description A collection of novel tools for generating species distribution and abundance models (SDM) that are dynamic through both space and time. These highly flexible functions incorporate spatial and temporal aspects across key SDM stages; including when cleaning and filtering species occurrence data, generating pseudo-absence records, assessing and correcting sampling biases and autocorrelation, extracting explanatory variables and projecting distribution patterns. Throughout, functions utilise Google Earth Engine and Google Drive to minimise the computing power and storage demands associated with species distribution modelling at high spatio-temporal resolution.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports dplyr, googledrive, lubridate, magrittr, reticulate, rgee, stats, terra, tidyr, grDevices, graphics, methods, utils, sf, readr

Depends R (>= 3.5.0)

Suggests ape, CoordinateCleaner, covr, gargle, gbm, ggplot2, knitr, magick, matrixStats, rmarkdown, spThin, stars, testthat (>= 3.0.0), viridis

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/r-a-dobson/dynamicSDM

BugReports https://github.com/r-a-dobson/dynamicSDM/issues

NeedsCompilation no

Author Rachel Dobson [aut, cre, ctb] (<https://orcid.org/0000-0003-3990-267X>), Andy J. Challinor [aut, ctb] (<https://orcid.org/0000-0002-8551-6617>), Robert A. Cheke [aut, ctb] (<https://orcid.org/0000-0002-7437-1934>), Stewart Jennings [aut, ctb] (<https://orcid.org/0000-0002-1267-8623>),

Stephen G. Willis [aut, ctb] (<https://orcid.org/0000-0002-8656-5808>),
Martin Dallimer [aut, ctb] (<https://orcid.org/0000-0001-8120-3309>)

Maintainer Rachel Dobson <eerdo@leeds.ac.uk>

Repository CRAN

Date/Publication 2023-10-18 06:20:02 UTC

R topics documented:

brt_fit *Fit boosted regression tree models to species distribution or abundance data.*

Description

Fit gradient boosting boosted regression tree models to species distribution and abundance data and associated dynamic explanatory variables.

Usage

```
brt_fit(
  occ.data,
  response.col,
  varnames,
  distribution,
  block.col,
  weights.col,
  test.data,
  interaction.depth,
  n.trees = 5000,
  shrinkage = 0.001
)
```

Arguments

occ.data	a data frame, the data to fit boosted regression tree models to, containing columns for model response and explanatory variable data. If required, occ.data should contain block.col and weights.col columns too.
response.col	a character string, the name of the column in occ.data containing response variable column.
varnames	a character vector, the names of the columns containing model explanatory variables in occ.data.
distribution	a character string, the model distribution family to use, such as gaussian, poisson or bernoulli.
block.col	optional; a character string, the name of the column in occ.data containing spatio-temporal block numbers for occ.data splitting. See details for more information.
weights.col	a character string, the name of the column in occ.data containing spatio-temporal sampling effort weights to be used in the model fitting process.
test.data	optional; a data frame, the testing dataset for optimising interaction.depth when blocking is not used.
interaction.depth	optional; an integer specifying the maximum depth of each tree (i.e. highest level of variable interactions allowed). Default optimises depth between 1 and 4.

n.trees	optional; an integer, the number of trees in boosted regression tree models. Default is 5000.
shrinkage	optional; an integer, the shrinkage parameter applied to each tree in the boosted regression tree expansion. Also known as the learning rate. Default is 0.001.

Details

This function calculates a gradient boosting gbm object for the response and explanatory variable data provided, using the gbm R package (Greenwell et al., 2019).

Key functionality for dynamic SDMs within brt_fit() includes:

- Optimise interaction.depth

If interaction.depth is not given, then brt_fit() will vary the interaction depth parameter between 1 (an additive model) and 4 (four-way interaction model). For each interaction.depth value, model performance is measured by calculating the root-mean-square error of model predictions compared to actual values in the testing data. The interaction.depth value that results in the lowest root-mean-square error is used when fitting the returned model.

The model testing dataset used can either be given using test.data or block.col (expanded on below).

- Split by spatio-temporal blocks to account for spatial and temporal autocorrelation

If block.col is specified, then each unique block is excluded in a jack-knife approach following Bagchi et al., (2013). This approach uses each block as the model testing dataset in numerical order, whilst all other block.col blocks are used as training data for the boosted regression tree model.

In this case, the function returns a list of fitted boosted regression tree models equal to the length of unique blocking categories in block.col.

If block.col is not given, models are fit to all occ.data and a single gbm model is returned.

- Weighted by spatio-temporal sampling effort

If weights.col is specified, records are weighted by their associated value in this column when model fitting. For instance, the user may wish to down weigh the importance of records collected at oversampled sites and times when fitting models, and vice versa, to account for spatio-temporal biases in occurrence records(Stolar and Nielsen, 2015) .

Value

Returns a gbm model object or list of gbm model objects.

References

Bagchi, R., Crosby, M., Huntley, B., Hole, D. G., Butchart, S. H. M., Collingham, Y., Kalra, M., Rajkumar, J., Rahmani, A. & Pandey, M. 2013. Evaluating the effectiveness of conservation site networks under climate change: accounting for uncertainty. Global Change Biology, 19, 1236-1248.

Greenwell, B., Boehmke, B., Cunningham, J., & GBM Developers. 2019. Package 'gbm'. R package version, 2.

Stolar, J. & Nielsen, S. E. 2015. Accounting For Spatially Biased Sampling Effort In Presence-Only Species Distribution Modelling. Diversity And Distributions, 21, 595-608.

Examples

```
data("sample_explan_data")

split <- sample(c(TRUE, FALSE),
                replace=TRUE,
                nrow(sample_explan_data),
                prob = c(0.75, 0.25))

training <- sample_explan_data[split, ]
testing <- sample_explan_data[!split, ]

brt_fit(
 occ.data = training,
 test.data = testing,
 response.col = "presence.absence",
 distribution = "bernoulli",
 varnames = colnames(training)[14:16],
 interaction.depth = 2
)
```

---

convert_gbif *Reformats GBIF data into* dynamicSDM *data frame*

Description

Function converts GBIF occurrence records into the format required for dynamicSDM functions.

Usage

```
convert_gbif(gbif.df)
```

Arguments

gbif.df	a data frame, the direct output from GBIF occurrence record download.

Details

For most dynamicSDM functions, an occurrence data frame with record co-ordinate columns labelled "x" and "y" with numeric columns for record "day", "month" and "year" are required. This function takes the input data frame and returns a reformatted data frame suitable for direct input into dynamicSDM functions.

Value

Returns data frame correctly formatted for input into dynamicSDM functions.

Examples

```
data(sample_occ_data)
converted <- convert_gbif(sample_occ_data)
```

`dynamic_proj` *Project species distribution and abundance models onto dynamic environmental covariates.*

Description

Projects fitted species distribution and abundance models onto projection covariates for each date given.

Usage

```
dynamic_proj(
  dates,
  projection.method,
  local.directory,
  drive.folder,
  user.email,
  sdm.mod,
  sdm.thresh,
  sdm.weight,
  sam.mod,
  sam.weight,
  save.directory,
  save.drive.folder,
  cov.file.type,
  prj = "+proj=longlat +datum=WGS84",
  proj.prj,
  spatial.mask
)
```

Arguments

`dates`	a character string, vector of dates in format "YYYY-MM-DD".
`projection.method`	a character string or vector, the method or methods to project distribution and abundance onto projection covariates. Options include `proportional`, `binary`, `abundance` and `stacked`. See details for more information.
`local.directory`	optional; a character string, the path to a local directory to read projection covariate data frames from.

drive.folder	optional; a character string, the Google Drive folder to read projection covariate data frames from. Folder must be uniquely named within Google Drive. Do not provide path.
user.email	optional; a character string, user email for initialising Google Drive. Required if drive.folder or save.drive.folder used.
sdm.mod	optional; a model object or list of model objects fitted to species distribution data.
sdm.thresh	optional; a numeric value, the threshold to convert projected distribution suitability into binary presence-absence. Default 0.5. Required if projection.method is "binary" or "stacked".
sdm.weight	optional; a numeric string, weights given to each sdm.mod model projection, given in the same order as the sdm.mod list. Default is equal weighting to all models.
sam.mod	optional; a model object or list of model objects fitted to species abundance data.
sam.weight	optional; a numeric string, weights given to each sdm.mod model projection, given in the same order as the sam.mod list. Default is equal weighting to all models.
save.directory	optional; a character string, path to local directory to save projection rasters to.
save.drive.folder	optional; a character string, Google Drive folder to save projection rasters to. Folder must be uniquely named within Google Drive. Do not provide path.
cov.file.type	a character string, the type of file that contains projection covariates. One of: "tif" (raster stack) or csv(data frame).
prj	a character string, the coordinate reference system of input projection covariates. Default is "+proj=longlat +datum=WGS84".
proj.prj	a character string, the coordinate reference system desired for output projection rasters. Default is assumed to be the same as prj.
spatial.mask	an object of class SpatRaster orsf polygon, representing a mask in which NA cells in the mask layer are removed from the projection covariates.

Details

Function projects a model object or list of model objects onto projection covariate data frames for each projection date given.

Value

Exports projection rasters for each projection date to user-specified Google Drive folder or local directory.

Projection covariate input

- Data frames: if cov.file.type = csv, then projection covariates must be saved "csv" files in the drive.folder or local.directory given. Here, they must be unique in containing the relevant projection date in YYYY-MM-DD format. For instance, two or more csv files saved

within the Google Drive folder or local directory that contain the projection date will result in function error. Additionally, column names of projection covariate data frames must match the explanatory variable names that fitted models are trained on.

- Raster stacks: if `cov.file.type = tif`, then projection covariates must be saved "tif" files, similarly named and formatted as above. Raster layer names must match the explanatory variable names that fitted models are trained on.

Note: It is important to state the coordinate reference system projection of covariates using argument `prj`.

Model input

When multiple models are provided in `sdm.mod` or `sam.mod`, the function projects each model onto the projection covariates and takes the average value across all model projections. If `sdm.weight` or `sam.weight` is specified, then the weighted average of model projections is returned. For example, this could be used to down weigh projections by poorly performing models in an ensemble (Ara?jo and New, 2007).

Projection output

- proportional: Projects `sdm.mod` model objects onto projection covariates for each date, exporting rasters for projected distribution suitability, a continuous measure between 0 (least suitable) and 1 (most suitable).
- binary: Projects `sdm.mod` onto projection covariates for each date, exporting rasters for projected binary presence (1) or absence (0), derived from distribution suitability using user-specified threshold (`sdm.thresh`) or default threshold of 0.5 (Jim?nez-Valverde And Lobo, 2007).
- abundance: Projects `sam.mod` onto projections covariates for each date, exporting rasters for projected abundance in the units that `sam.mod` were fitted onto.
- stacked: Follows the binary projection method and then projects abundance onto only binary presence (1) cells using the abundance projection method.

Projections are output as rasters. These can be reprojected to a different coordinate reference system using argument `proj.prj`.

One or both of `save.drive.folder` and `save.directory` are required to specify where projection rasters are to be saved.

Google Drive compatibility

If `drive.folder` or `save.drive.folder` given, please ensure the folder name is unique within your Google Drive. Do not provide the path if the folder is nested within others.

If one of `drive.folder` or `save.drive.folder` are used then `user.email` is required to access the appropriate Google Drive user account. This requires users to have installed R package `googledrive` and initialised Google Drive with valid log-in credentials. Please follow instructions on https://googledrive.tidyverse.org/.

References

Araujo, M. B. & New, M. 2007. Ensemble Forecasting Of Species Distributions. Trends In Ecology & Evolution, 22, 42-47.

Jimenez-Valverde, A. & Lobo, J. M. 2007. Threshold Criteria For Conversion Of Probability Of Species Presence To Either-Or Presence-Absence. Acta Oecologica, 31, 361-369.

Examples

```
# Read in data
data("sample_explan_data")

# Set variable names
variablenames<-c("eight_sum_prec","year_sum_prec","grass_crop_percentage")

model <- brt_fit(sample_explan_data,
                 response.col = "presence.absence",
                 varnames = variablenames,
                 interaction.depth = 1,
                 distribution = "bernoulli",
                 n.trees = 1500)

data(sample_cov_data)
utils::write.csv(sample_cov_data,file=paste0(tempdir(),"/2018-04-01_covariates.csv"))

dynamic_proj(dates = "2018-04-01",
            projection.method = c("proportional"),
            local.directory = tempdir(),
            cov.file.type = "csv",
            sdm.mod = model,
            save.directory = tempdir())
```

---

dynamic_proj_covariates

Combine explanatory variable rasters into covariates for each projection date.

Description

Explanatory variable rasters are imported, resampled to a given spatial resolution and extent, stacked and then exported as a covariate data frame or raster stack for each projection date.

Usage

```
dynamic_proj_covariates(
  dates,
  varnames,
```

```
  drive.folder,
  user.email,
  local.directory,
  spatial.ext,
  spatial.mask,
  spatial.res.degrees,
  resample.method,
  cov.file.type,
  prj = "+proj=longlat +datum=WGS84",
  cov.prj,
  save.directory,
  save.drive.folder,
  static.rasters,
  static.varnames,
  static.resample.method,
  static.moving.window.matrix,
  static.GEE.math.fun
)
```

Arguments

dates	a character string, vector of dates in format "YYYY-MM-DD".
varnames	a character string, the unique names for each explanatory variable.
drive.folder	optional; a character string or vector, Google Drive folder or folders to read projection covariate rasters from. Folder must be uniquely named within Google Drive. Do not provide path.
user.email	optional; a character string, user email for initialising Google Drive. Required if drive.folder or save.drive.folder used.
local.directory	optional; a character string or vector, path to local directory or directories to read projection covariate rasters from.
spatial.ext	optional; the spatial extent to crop explanatory variable rasters to. Object of class SpatExtent, SpatRaster, an sf polygon or numeric vector listing xmin, xmax, ymin and ymax in order.
spatial.mask	an object of class Raster, sf or Spatial, representing a mask in which NA cells in the mask layer are removed from the projection covariates.
spatial.res.degrees	optional; a numeric value, the spatial resolution in degrees for projection rasters to be resampled to. Required if spatial.ext given.
resample.method	a character string or vector length of varnames, specifying resampling method to use. One of near and bilinear. See details for more information.
cov.file.type	a character string, the type of file to export projection covariates as. One of: tif (SpatRaster with multiple layers) or csv(data frame).
prj	a character string, the coordinate reference system desired for projection covariates. Default is "+proj=longlat +datum=WGS84".

`cov.prj` a character string, the coordinate reference system desired for output projection covariates. Default is assumed to be the same as `prj`.

`save.directory` optional; a character string, path to local directory to save projection covariates to.

`save.drive.folder` optional; a character string, Google Drive folder to save projection covariates to. Folder must be uniquely named within Google Drive. Do not provide path.

`static.rasters` a RasterStack of one or more rasters to be added to covariates for each date.

`static.varnames` a character string or vector, the unique names for each explanatory variable in order of rasters in `static.raster` stack.

`static.resample.method` a character string or vector length of `static.varnames`, specifying resampling method to use on static rasters provided. One of `near` and `bilinear`. See details for more information..

`static.moving.window.matrix` optional; a matrix of weights with an odd number of sides, representing the spatial neighbourhood of cells ("moving window") to calculate `GEE.math.fun` across from record co-ordinate. See details for more information.

`static.GEE.math.fun` optional; a character string, the mathematical function to compute across the specified spatial matrix for each cell in `spatial.ext`

Value

Exports combined covariates in "csv" or "tif" file for each projection date to the local directory or Google Drive folder.

Input variable rasters

For each projection date, the rasters for each explanatory variable are imported from a local directory or Google Drive folder.

Such rasters should be uniquely named "tif" files within the directory or drive folder and contain the variable name (as stated in `varnames`) and projection date in format "YYYY-MM-DD". If more than one "tif" file in the Google Drive folder or local directory matches the projection date and explanatory variable name, then the function will error.

Processing rasters

If required, rasters are cropped and resampled to the same spatial extent and resolution. If `spatial.mask` is given, then cells with NA in this mask layer are removed from the returned projection covariates. See `terra::mask()` in R package `terra` for details (Hijmans et al., 2022).

Rasters are then stacked and reprojected if `cov.prj` is different to `prj`.

Note: if explanatory variable rasters are not of the same spatial resolution and extent, then the function will error. Resample methods (`resample.method`) include:

- `near`: Each cell acquires the value of its nearest neighbour cell in the original raster. This is typically used for categorical variables.

- `bilinear`: the distance-weighted average of the four nearest cells are used to estimate a new cell value. This is typically used for continuous variables.

If only one `resample.method` is given, but these are more than one explanatory variables, the same `resample.method` is used for all.

Output covariates

The raster stacks are then converted into data frames or remain as raster stacks depending on `cov.file.type`. Column names or raster layer names will be the unique explanatory variable names (`varnames`). These are exported to the local directory or Google Drive folder with file names containing the relevant projection date in "YYYY-MM-DD" format.

Google Drive compatibility

If `drive.folder` or `save.drive.folder` given, please ensure the folder name is unique within your Google Drive. Do not provide the path if the folder is nested within others.

If one of `drive.folder` or `save.drive.folder` are used then user.email is required to access the appropriate Google Drive user account. This requires users to have installed R package `googledrive` and initialised Google Drive with valid log-in credentials. Please follow instructions on https://googledrive.tidyverse.org/.

Static rasters

If static datasets are to be added into the dynamic projection covariates, then five arguments are available to specify their inclusion.

Please provide the static rasters in RasterStack format, specifying any spatial buffering needed (using `static.moving.window.matrix` and `static.GEE.math.fun` as described below).

The resample method or methods for rasters within the static raster stack need to be specified too using `static.resample.method`. Please also provide `static.varnames` to name the static variables in the covariate data exported.

#' # Spatial buffering of static rasters (optional)

Using the `focal` function from `terra` R package (Hijmans et al., 2022), `GEE.math.fun` is calculated across the spatial buffer area from the record co-ordinate. The spatial buffer area used is specified by the argument `moving.window.matrix`, which dictates the neighbourhood of cells surrounding the cell containing the occurrence record to include in this calculation.

See function `get_moving_window()` to generate appropriate `moving.window.matrix`.

Mathematical function

`GEE.math.fun` specifies the mathematical function to be calculated over the spatial buffered area and temporal period. Options are limited to Google Earth Engine ImageCollection Reducer functions (https://developers.google.com/earth-engine/apidocs/) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

References

Hijmans, R.J., Bivand, R., Forner, K., Ooms, J., Pebesma, E. and Sumner, M.D., 2022. Package 'terra'. Maintainer: Vienna, Austria.

Examples

```
data("sample_extent_data")

# Set extraction variables
projectiondates <- dynamic_proj_dates("2018-01-01", "2018-12-01", interval = 3,interval.level =
"month")
variablenames <- c("eight_sum_prec", "year_sum_prec")
spatial.res.metres <- 500
cov_resolution <- 0.05

# Get Google Drive email
user.email<-as.character(gargle::gargle_oauth_sitrep()$email)

extract_dynamic_raster(dates=projectiondates,
                       datasetname = "UCSB-CHG/CHIRPS/DAILY",
                       bandname="precipitation",
                       user.email = user.email,
                       spatial.res.metres = spatial.res.metres,
                       GEE.math.fun = "sum",
                       temporal.direction = "prior",
                       temporal.res = 56,
                       spatial.ext = sample_extent_data,
                       varname = variablenames[1],
                       save.directory = tempdir())


extract_dynamic_raster(dates=projectiondates,
                      datasetname = "UCSB-CHG/CHIRPS/DAILY",
                      bandname="precipitation",
                      user.email = user.email,
                      spatial.res.metres = spatial.res.metres,
                      GEE.math.fun = "sum",
                      temporal.direction = "prior",
                      temporal.res = 364,
                      spatial.ext = sample_extent_data,
                      varname = variablenames[2],
                      save.directory = tempdir())

dynamic_proj_covariates(dates = projectiondates,
                        varnames = variablenames,
                        local.directory = tempdir(),
                        spatial.ext = sample_extent_data,
                        spatial.mask = sample_extent_data,
                        spatial.res.degrees = cov_resolution,
                        resample.method = c("bilinear","bilinear"),
```

```
cov.file.type = "csv",
prj="+proj=longlat +datum=WGS84",
save.directory = tempdir())
```

`dynamic_proj_dates` *Generate vector of dates for dynamic projections*

Description

Creates a vector of dates at regular intervals between two given dates.

Usage

```
dynamic_proj_dates(startdate, enddate, interval.level, interval)
```

Arguments

`startdate`	a character string, the start date in format "YYYY-MM-DD".
`enddate`	a character string, the end date in format "YYYY-MM-DD".
`interval.level`	a character string, the time-step of intervals. One of `day`,`week`, `month` or `year`: can be abbreviated.
`interval`	a numeric value, the length of interval in `interval.level` units to generate between the start and end date.

Details

Function returns a vector of dates between `start.date` and `end.date` at given interval size.

Value

Vector of dates between start date and end date split at regular intervals.

Examples

```
dynamic_proj_dates(
 startdate = "2000-01-01",
 enddate = "2001-01-01",
 interval.level = "month",
 interval = 2
)
```

dynamic_proj_GIF *Create GIF of dynamic species distribution and abundance projections*

Description

Plots dynamic species distribution and abundance projections through time and combines images into a GIF.

Usage

```
dynamic_proj_GIF(
  dates,
  projection.type,
  drive.folder,
  user.email,
  local.directory,
  save.drive.folder,
  save.directory,
  width = 10,
  height = 10,
  legend.max,
  legend.min,
  legend.name,
  file.name,
  borders = FALSE,
  border.regions,
  border.colour = "black",
  colour.palette.custom,
  colour.palette
)
```

Arguments

dates	a character vector , projection dates in format "YYYY-MM-DD".
projection.type	a character string, the type of distribution or abundance projection to plot. One of proportional, binary, abundance and stacked.
drive.folder	optional; a character string, the Google Drive folder to read projection rasters from. Folder must be uniquely named within Google Drive. Do not provide path.
user.email	optional; a character string, user email for initialising Google Drive. Required if drive.folder or save.drive.folder used.
local.directory	optional; a character string, the path to local directory to read projection rasters from.

`save.drive.folder`
: optional; a character string, Google Drive folder to save GIF to. Folder must be uniquely named within Google Drive. Do not provide path.

`save.directory`
: optional; a character string, path to local directory to save GIF to.

`width`
: optional; a numeric value, the GIF width in inches Default = 480.

`height`
: optional; a numeric value, the GIF height in inches Default = 480.

`legend.max`
: optional; a numeric value, the maximum limit of legend values to standardise across projections.

`legend.min`
: optional; a numeric value, the minimum limit of legend values to standardise across projections.

`legend.name`
: optional; a character string, the name for the legend title. Default = projection.type.

`file.name`
: optional, a character string, the name for the output GIF file. Default = `projection.type`.

`borders`
: a logical indicating whether to add country borders to map. Default = `FALSE`.

`border.regions`
: optional; a character string or vector, the region or regionss for which to add map borders. Required if `borders` = `TRUE`.

`border.colour`
: optional; a character vector, the colour for plotted map borders. Default = `black`.

`colour.palette.custom`
: optional; a character string or vector, the colours to use as plot colour palette.

`colour.palette`
: optional; a character string, the colormap option to use from `viridis`. See details for colour palette options.

Details

Function reads in projection rasters for each date. These are plotted using `ggplot2` and combined into a Graphics Interchange Format (GIF).

Value

Exports GIF to Google Drive folder or local directory.

Import projection rasters

Projection rasters for each date must be "tif" files that are uniquely named with the date in format "YYYY-MM-DD" and `projection.type`. If more than one file name matches the date and `projection.type`, the function will error.

Google Drive compatibility

If `drive.folder` or `save.drive.folder` is given, please ensure the folder name is unique within your Google Drive. Do not provide the path if the folder is nested within others.

If one of `drive.folder` or `save.drive.folder` are used then `user.email` is required to access the appropriate Google Drive user account. This requires users to have installed R package `googledrive` and initialised Google Drive with valid log-in credentials. Please follow instructions on https://googledrive.tidyverse.org/.

Options for colour palettes using `viridis` are illustrated at: https://ggplot2.tidyverse.org/reference/scale_viridis.html. Available options include: "magma" (or "A"), "inferno" (or "B"), "plasma" (or "C"), "viridis" (or "D", the default option), "cividis" (or "E"), "rocket" (or "F"), "mako"(or "G") and "turbo" (or "H").

References

Wickham, H., and Chang, W, 2016. Package 'ggplot2'. Create elegant data visualisations using the grammar of graphics. Version, 2(1), pp.1-189.

Examples

```
proj_dates <- dynamic_proj_dates(startdate = "2018-01-01",
                                 enddate = "2018-12-01",
                                 interval = 3,
                                 interval.level = "month")

# Generate fake proportional projection SpatRasters
proj1 <- terra::rast(ncols=22, nrows=25)
proj1 <- terra::setValues(proj1, sample(0:100, terra::ncell(proj1), replace = TRUE)/100)

proj2 <- terra::rast(ncols=22, nrows=25)
proj2 <- terra::setValues(proj2, sample(0:100, terra::ncell(proj2), replace = TRUE)/100)

proj3 <- terra::rast(ncols=22, nrows=25)
proj3 <- terra::setValues(proj3, sample(0:100, terra::ncell(proj3), replace = TRUE)/100)

proj4 <- terra::rast(ncols=22, nrows=25)
proj4 <- terra::setValues(proj4, sample(0:100, terra::ncell(proj4), replace = TRUE)/100)

# Save sample projection rasters to replicate output from `dynamic_proj()`
terra::writeRaster(proj1,
                   filename = paste0(tempdir(),
                   "/",
                   paste0(proj_dates[1], "_proportional.tif")),
                   overwrite = TRUE)
terra::writeRaster(proj2,
                   filename = paste0(tempdir(),
                   "/",
                   paste0(proj_dates[2], "_proportional.tif")),
                   overwrite = TRUE)
terra::writeRaster(proj3,
                   filename = paste0(tempdir(),
                   "/",
                   paste0(proj_dates[3], "_proportional.tif")),
                   overwrite = TRUE)
terra::writeRaster(proj4,
                   filename = paste0(tempdir(),
                   "/",
                   paste0(proj_dates[4], "_proportional.tif")),
                   overwrite = TRUE)
```

```
dynamic_proj_GIF(dates = proj_dates,
                 projection.type = "proportional",
                 local.directory = tempdir(),
                 save.directory = tempdir())
```

`extract_buffered_coords` *Extract spatially buffered and temporally dynamic explanatory variable data for occurrence records.*

Description

For each species occurrence record co-ordinate and date, spatially buffered and temporally dynamic explanatory data are extracted using Google Earth Engine.

Usage

```
extract_buffered_coords(
  occ.data,
  datasetname,
  bandname,
  spatial.res.metres,
  GEE.math.fun,
  moving.window.matrix,
  user.email,
  save.method,
  varname,
  temporal.res,
  temporal.level,
  temporal.direction,
  categories,
  save.directory,
  agg.factor,
  prj = "+proj=longlat +datum=WGS84",
  resume = TRUE
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
`datasetname`	a character string, the Google Earth Engine dataset to extract data from.
`bandname`	a character string, the Google Earth Engine dataset bandname to extract data for.

`spatial.res.metres`
: a numeric value, the spatial resolution in metres for data extraction.

`GEE.math.fun`
: a character string, the mathematical function to compute across the specified spatial matrix and period for each record.

`moving.window.matrix`
: a matrix of weights with an odd number of sides, representing the spatial neighbourhood of cells ("moving window") to calculate `GEE.math.fun` across from record co-ordinate. See details for more information.

`user.email`
: a character string, user email for initialising Google Drive.

`save.method`
: a character string, the method used to save extracted variable data. One of `split` or `combined`: can be abbreviated. See details.

`varname`
: optional; a character string, a unique name for the explanatory variable. Default varname is "bandname_temporal.res_temporal.direction_GEE.math.fun_buffered".

`temporal.res`
: optional; a numeric value, the temporal resolution in days to extract data and calculate `GEE.math.fun` across from occurrence record date.

`temporal.level`
: a character string, the temporal resolution of the explanatory variable data. One of `day`, `month` or `year`: can be abbreviated. Default; `day`.

`temporal.direction`
: optional; a character string, the temporal direction for extracting data across relative to the record date. One of `prior` or `post`: can be abbreviated.

`categories`
: optional; a character string, the categories to use in calculation if data are categorical. See details for more information.

`save.directory`
: a character string, path to a local directory to save extracted variable data to.

`agg.factor`
: optional;a postive integer, the aggregation factor expressed as number of cells in each direction. See details.

`prj`
: a character string, the coordinate reference system of `occ.data` coordinates.Default is "+proj=longlat +datum=WGS84".

`resume`
: a logical indicating whether to search `save.directory` and return to previous progress. Only possible if save.method = `split` has previously and currently been employed. Default = TRUE.

Details

For each individual species occurrence record co-ordinate and date, this function extracts data for a given band within a Google Earth Engine dataset across a user-specified spatial buffer and temporal period and calculates a mathematical function on such data.

Value

Returns details of successful explanatory variable extractions.

Temporal dimension

If `temporal.res` and `temporal.direction` are not given, the function extracts explanatory variable data for all of the cells surrounding and including the cell containing the occurrence record co-ordinates.

If temporal.res and temporal.direction is given, the function extracts explanatory variable data for which GEE.math.fun has been first calculated over this period in relation to the occurrence record date.

Spatial dimension

Using the focal function from terra R package (Hijmans et al., 2022), GEE.math.fun is calculated across the spatial buffer area from the record co-ordinate. The spatial buffer area used is specified by the argument moving.window.matrix, which dictates the neighbourhood of cells surrounding the cell containing the occurrence record to include in this calculation.

See function get_moving_window() to generate appropriate moving.window.matrix.

Mathematical function

GEE.math.fun specifies the mathematical function to be calculated over the spatial buffered area and temporal period. Options are limited to Google Earth Engine ImageCollection Reducer functions (https://developers.google.com/earth-engine/apidocs/) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

Categorical data

When explanatory variable data are categorical (e.g. land cover classes), argument categories can be used to specify the categories of importance to the calculation. The category or categories given will be converted in a binary representation, with "1" for those listed, and "0" for all others in the dataset. Ensure that the GEE.math.fun given is appropriate for such data. For example, the sum of suitable land cover classified cells across the "moving window" from the species occurrence record co-ordinates.

Categorical data and temporally dynamic variables

Please be aware, if specific categories are given (argument categories) when extracting categorical data, then temporal buffering cannot be completed. The most recent categorical data to the occurrence record date will be used for spatial buffering.

If specific categories are not given when extracting from categorical datasets, be careful to choose appropriate mathematical functions for such data. For instance, "first" or "last" may be more relevant that "sum" of land cover classification numbers.

Temporal level to extract data at:

temporal.level states the temporal resolution of the explanatory variable data and improves the speed of extract_buffered_coords() extraction. For example, if the explanatory data represents an annual variable, then all record co-ordinates from the same year can be extracted from the same buffered raster, saving computation time. However, if the explanatory data represents a daily variable, then only records from the exact same day can be extracted from the same raster. For the former, temporal.level argument should be year and for the latter, temporal.level should be day.

Aggregation factor

agg.factor given represents the factor to aggregate RasterLayer data with function aggregate in terra R package (Hijmans et al., 2022). Aggregation uses the GEE.math.fun as the function. Following aggregation spatial buffering using the moving window matrix occurs. This is included to minimise computing time if data are of high spatial resolution and a large spatial buffer is needed. Ensure to calculate get_moving_window() with the spatial resolution of the data post-aggregation by this factor.

Google Earth Engine

extract_buffered_coords() requires users to have installed R package rgee (Aybar et al., 2020) and initialised Google Earth Engine with valid log-in credentials. Please follow instructions on the following website https://cran.r-project.org/package=rgee

- datasetname must be in the accepted Google Earth Engine catalogue layout (e.g. "MODIS/006/MCD12Q1" or "UCSB-CHG/CHIRPS/DAILY")
- bandname must be as specified under the dataset in the Google Earth Engine catalogue (e.g. "LC_Type5", "precipitation"). For datasets and band names, see https://developers.google.com/earth-engine/datasets.

Google Drive

extract_buffered_coords() also requires users to have installed the R package googledrive(D'Agostino McGowan and Bryan, 2022) and initialised Google Drive with valid log-in credentials, which must be stated using argument user.email. Please follow instructions on https://googledrive.tidyverse.org/ for initialising the googledrive package.

Note: When running this function a folder labelled "dynamicSDM_download_bucket" will be created in your Google Drive. This will be emptied once the function has finished running and output rasters will be found in the save.drive.folder or save.directory specified.

Exporting extracted data

For save.method = combined, the function with save "csv" files containing all occurrence records and associated values for the explanatory variable.

For save.method = split, the function will save individual "csv" files for each record with each unique period of the given temporal.level (e.g. each year, each year and month combination or each unique date).

split protects users if internet connection is lost when extracting data for large occurrence datasets. The argument resume can be used to resume to previous progress if connection is lost.

References

Aybar, C., Wu, Q., Bautista, L., Yali, R. and Barja, A., 2020. rgee: An R package for interacting with Google Earth Engine. Journal of Open Source Software, 5(51), p.2272.

D'Agostino McGowan L., and Bryan J., 2022. googledrive: An Interface to Google Drive. https://googledrive.tidyverse.org, https://github.com/tidyverse/googledrive.

Hijmans, R.J., Bivand, R., Forner, K., Ooms, J., Pebesma, E. and Sumner, M.D., 2022. Package 'terra'. Maintainer: Vienna, Austria.

Examples

```
data(sample_filt_data)



user.email<-as.character(gargle::gargle_oauth_sitrep()$email)

matrix<-get_moving_window(radial.distance = 10000,
                          spatial.res.degrees = 0.05,
                          spatial.ext = sample_extent_data)

extract_buffered_coords(occ.data = sample_filt_data,
                      datasetname = "MODIS/006/MCD12Q1",
                      bandname = "LC_Type5",
                      spatial.res.metres = 500,
                      GEE.math.fun = "sum",
                      moving.window.matrix=matrix,
                      user.email = user.email,
                      save.method ="split",
                      temporal.level = "year",
                      categories = c(6,7),
                      agg.factor = 12,
                      varname = "total_grass_crop_lc",
                      save.directory = tempdir()
)
```

---

extract_buffered_raster

Extract spatially buffered and temporally dynamic rasters of explanatory variable data.

Description

Extract rasters for spatially buffered and temporally dynamic explanatory variables at each projection date using Google Earth Engine.

Usage

```
extract_buffered_raster(
  dates,
  spatial.ext,
  datasetname,
  bandname,
  temporal.level,
  spatial.res.metres,
  GEE.math.fun,
```

```
  moving.window.matrix,
  user.email,
  varname,
  temporal.res,
  temporal.direction,
  categories,
  save.directory,
  agg.factor,
  save.drive.folder,
  resume = TRUE
)
```

Arguments

dates	a character string, vector of dates in format "YYYY-MM-DD".
spatial.ext	the spatial extent for the extracted raster. Object from which extent can be extracted of class SpatExtent, SpatRaster, sf polygon or numeric vector listing xmin, xmax, ymin and ymax in order.
datasetname	a character string, the Google Earth Engine dataset to extract data from.
bandname	a character string, the Google Earth Engine dataset bandname to extract data for.
temporal.level	a character string indicating the temporal resolution of the remote-sensing dataset (datasetname). One of day, month or year: can be abbreviated. Default; day.
spatial.res.metres	a numeric value, specifying the spatial resolution in metres of the raster to be extracted.
GEE.math.fun	a character string, the mathematical function to compute across the specified period and spatial buffer from each projection date and cell.
moving.window.matrix	a matrix of weights with an odd number of sides to specify spatial neighbourhood of cells ("moving window") to calculate GEE.math.fun across for each cell in spatial.ext. See details for more information.
user.email	a character string, user email for initialising Google Drive.
varname	optional; a character string, the unique name for the explanatory variable. Default varname is “bandname_temporal.res_temporal.direction_ GEE.math.fun_buffered_raster".
temporal.res	optional; a numeric value, the temporal resolution in days prior or post each projection date to calculate GEE.math.fun across.
temporal.direction	optional; a character string, the temporal direction for extracting dynamic variable data across relative to each projection date given. One of prior or post: can be abbreviated.
categories	optional; a character string, the categories to use in the calculation if data are categorical. See details for more information.
save.directory	optional; a character string, path to local directory to save extracted rasters to.
agg.factor	optional;a postive integer, the aggregation factor expressed as number of cells in each direction. See details.

`save.drive.folder`
: optional; a character string, Google Drive folder to save extracted rasters to. Folder must be uniquely named within Google Drive. Do not provide path.

`resume`
: a logical indicating whether to search `save.directory` or `save.drive.folder` and return to previous progress through projection dates.Default = TRUE.

Details

For each projection date, this function downloads rasters at a given spatial extent and resolution for spatially buffered and temporally dynamic explanatory variables. Rasters can be saved directly to Google Drive or a local directory. These rasters can be combined to create projection covariate data frames for projecting dynamic species distribution and abundance at high spatiotemporal resolution.

Value

Returns details of successful explanatory variable raster extractions for each projection date.

Temporal dimension

If `temporal.res` and `temporal.direction` are not given, explanatory variable data for all cells within `spatial.ext` are extracted. If `temporal.res` and `temporal.direction` are given, explanatory variable data for all cells within `spatial.ext` are extracted, for which `GEE.math.fun` has been first calculated over the specified period in relation to the projection date (prior or post).

Categorical data and temporally dynamic variables

Please be aware, if specific categories are given (argument `categories`) when extracting categorical data, then temporal buffering cannot be completed. The most recent categorical data to the occurrence record date will be used and spatial buffering will take place.

If, specific categories are not given when extracting from categorical datasets, be careful to choose appropriate mathematical functions for such data. For instance, "first" or "last" may be more relevant that "sum" of land cover classification numbers.

Spatial dimension

Using the `focal` function in `terra` R package (Hijmans et al., 2022), `GEE.math.fun` is calculated across the spatial buffer area from each cell in `spatial.ext`. The spatial buffer area used is defined by `moving.window matrix`, which dictates the neighbourhood of cells surrounding each cell in `spatial.ext` to include in the calculation. See get_moving_window.

Mathematical function

`GEE.math.fun` specifies the mathematical function to be calculated over the spatial buffered area and temporal period. Options are limited to Google Earth Engine ImageCollection Reducer functions (`https://developers.google.com/earth-engine/apidocs/`) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

Categorical data

If explanatory variable data are categorical (e.g. land cover classes), `categories` can be used to specify the categories of importance to the calculation. The category or categories given will be converted in a binary representation, with "1" for those listed, and "0" for all others in the dataset. Ensure that the `GEE.math.fun` given is appropriate for such data.

For example, this function could return the sum of suitable land cover classified cells in the "moving window" from each cell across spatial extent given.

Aggregation factor

`agg.factor` given represents the factor to aggregate `SpatRaster` data with function `aggregate` in `terra` R package (Hijmans et al., 2022). Aggregation uses the `GEE.math.fun` as the function. Following aggregation spatial buffering using the moving window matrix occurs. This is included to minimise computing time if data are of high spatial resolution and a large spatial buffer is needed. Ensure to calculate `get_moving_window()` with the spatial resolution of the data post-aggregation by this factor.

Google Earth Engine

`extract_buffered_raster()` requires users to have installed R package `rgee` (Aybar et al., 2020) and initialised Google Earth Engine with valid log-in credentials. Please follow instructions on the following website https://cran.r-project.org/package=rgee

- `datasetname` must be in the accepted Google Earth Engine catalogue layout (e.g. "MODIS/006/MCD12Q1" or "UCSB-CHG/CHIRPS/DAILY")
- `bandname` must be as specified under the dataset in the Google Earth Engine catalogue (e.g. "LC_Type5", "precipitation"). For datasets and band names, see https://developers.google.com/earth-engine/datasets.

Google Drive

`extract_buffered_raster()` also requires users to have installed the R package `googledrive`(D'Agostino McGowan and Bryan, 2022) and initialised Google Drive with valid log-in credentials, which must be stated using argument `user.email`. Please follow instructions on https://googledrive.tidyverse.org/ for initialising the `googledrive` package.

The `save.drive.folder` must be uniquely named within your Google Drive and do not provide the path.

Occasional rgee errors

As this function uses the `rgee` package to extract rasters from Google Earth Engine, below we outline occasional `rgee` errors that may occur when extracting rasters:

- "To avoid memory excess problems, ee_as_raster will not build Raster objects for large images"

This can be a sporadic error. It may be related to GEE server usage or internet connection at the time you tested the function. Try restarting your R session or try again at another time. Also, try clearing the files from the "dynamicSDM_download_bucket" in your Google Drive.

This error could also be due to an issue with your input `spatial.res.metres`. This function will extract rasters at all typical spatial resolutions of remote-sensing data and at global extents. If this error persists, please ensure you have not accidentally given an unrealistically high spatial resolution (e.g. `spatial.res.metres = 0.01` when you may be confusing the spatial resolution in degrees with metres).

- "Pixel type not supported: Type Long. Convert the image to a floating point type or a smaller integer type, for example, using ee.Image.toDouble()"

This error appears when `rgee` has been given an input that cannot be extracted. Some common causes include:

- Inappropriate `GEE.math.fun` for extracting categorical data.
- Dates or spatial extents that are not covered by the chosen GEE dataset. Remember to check whether the first projection date minus `temporal.res` is still within the temporal extent of the dataset.

References

Aybar, C., Wu, Q., Bautista, L., Yali, R. and Barja, A., 2020. rgee: An R package for interacting with Google Earth Engine. Journal of Open Source Software, 5(51), p.2272.

D'Agostino McGowan L., and Bryan J., 2022. googledrive: An Interface to Google Drive. https://googledrive.tidyverse.org, https://github.com/tidyverse/googledrive.

Hijmans, R.J., Bivand, R., Forner, K., Ooms, J., Pebesma, E. and Sumner, M.D., 2022. Package 'terra'. Maintainer: Vienna, Austria.

Examples

```
dates <- dynamic_proj_dates("2018-01-01", "2018-12-01", interval = 3,interval.level = "month")


data("sample_extent_data")
user.email<-as.character(gargle::gargle_oauth_sitrep()$email)

matrix<-get_moving_window(radial.distance = 10000,
                          spatial.res.degrees = 0.05,
                          spatial.ext = sample_extent_data)

extract_buffered_raster(dates = dates,
                        datasetname = "MODIS/006/MCD12Q1",
                        bandname="LC_Type5",
                        spatial.res.metres = 500,
                        GEE.math.fun = "sum",
                        moving.window.matrix = matrix,
                        user.email = user.email,
                        categories=c(6,7),
                        agg.factor = 12,
                        spatial.ext = sample_extent_data,
                        varname = "total_grass_crop_lc",
```

```
                        save.directory = tempdir())
```

`extract_coords_combine` *Combine extracted explanatory variable data for occurrence records into single data frame.*

Description

Combines the split output files from functions `extract_dynamic_coords()` and `extract_buffered_coords()` into single data frame containing all occurrence records and explanatory variables.

Usage

```
extract_coords_combine(
  varnames,
  local.directory,
  set_class = FALSE,
  col_classes
)
```

Arguments

`varnames`	a character string, the unique names for each explanatory variable.
`local.directory`	a character string or vector, the path to local directory or directories to read extracted explanatory data frames from.
`set_class`	a logical indicating whether to set the classes of each column in the data before merging to avoid error. See details for more information.
`col_classes`	optional; a named vector specifying the classes for columns within occ.data.

Details

When functions `extract_dynamic_coords()` and `extract_buffered_coords()` have been used to extract dynamic explanatory variables for occurrence records, the output for individual records and each variable will be split into separate "csv" files.

This function reads in these files and combines data into a single data frame containing each occurrence records and associated explanatory data from each variable.

To prevent error, the "csv" files must be uniquely named within the folder(s) and include an exact character match for the varnames provided. All "csv" files matching the varnames should have the same number and names of columns. This is the default output of`extract_dynamic_coords()` and `extract_buffered_coords()`.

Value

Returns a data frame containing all occurrence records with associated explanatory variable data.

Column classes

When co-ordinate data have been extracted using the `split` method in `extract_dynamic_coords()` and `extract_buffered_coords()`, sometimes the classes of columns in the separate exported data frames can vary. For instance, one split row may contain an NA value in one column, whilst another split contains a numerical value. When `extract_coords_combine()` attempts to read these in and bind them together, an error can occur due to the class mismatch (logical compared to numeric).

There are two arguments that can help to resolve this error:

- If `set_class = TRUE` and `col_classes` is given, then the column classes for each split data frame will be set as the classes in `col_classes`.
- If `set_class = TRUE` and `col_classes` is not given, then the column classes for each split data frame will be set by the classes of columns in the first read in csv. Be aware that this approach may be inaccurate and lead to parsing warnings, as it depends on the first split containing the correct classes (e.g. numeric not a logical in the example above).

Examples

```
data(sample_filt_data)


dynamicSDM::extract_dynamic_coords(
 occ.data = sample_filt_data,
 datasetname = "UCSB-CHG/CHIRPS/DAILY",
 bandname = "precipitation",
 spatial.res.metres = 10000,
 GEE.math.fun = "sum",
 temporal.direction = "prior",
 temporal.res = 56,
 save.method = "split",
 varname = "eightweekprec",
 save.directory = tempdir()
)
dynamicSDM::extract_dynamic_coords(
 occ.data = sample_filt_data,
 datasetname = "UCSB-CHG/CHIRPS/DAILY",
 bandname = "precipitation",
 spatial.res.metres = 10000,
 GEE.math.fun = "sum",
 temporal.direction = "prior",
 temporal.res = 364,
 save.method = "combined",
 varname = "annualweekprec",
 save.directory = tempdir()
)

extract_coords_combine(varnames = c("eightweekprec", "annualweekprec"),
                       local.directory = tempdir(),
                       set_class = TRUE,
                       col_classes = c(sapply(sample_filt_data,class),
                                       eightweekprec = "numeric",
```

```
                                annualweekprec="numeric"))
```

`extract_dynamic_coords` *Extract temporally dynamic explanatory variable data for occurrence records.*

Description

For each species occurrence record co-ordinate and date, temporally dynamic explanatory data are extracted using Google Earth engine

Usage

```
extract_dynamic_coords(
  occ.data,
  datasetname,
  bandname,
  spatial.res.metres,
  GEE.math.fun,
  save.method,
  temporal.res,
  temporal.direction,
  varname,
  resume = FALSE,
  save.directory
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
`datasetname`	a character string, the Google Earth Engine dataset to extract data from.
`bandname`	a character string, the Google Earth Engine dataset bandname to extract data for.
`spatial.res.metres`	a numeric value, the spatial resolution in metres for data extraction.
`GEE.math.fun`	a character string, the mathematical function to compute across the `temporal.res` period for each record.
`save.method`	a character string, the method used to save extracted variable data. One of `split` or `combined`: can be abbreviated. See details.
`temporal.res`	a numeric value, the temporal resolution in days to extract data and calculate `GEE.math.fun` across from each record's date.

temporal.direction
: a character string, the temporal direction for extracting data across relative to the record date. One of prior or post: can be abbreviated.

varname
: optional; a character string, the unique name for the explanatory variable. Default varname is "bandname_temporal.res_temporal.direction_ GEE.math.fun".

resume
: a logical indicating whether to search save.directory and start from previous progress by function. Only possible if save.method = split has been used.

save.directory
: a character string, the path to a local directory to save extracted variable data to.

Details

For each individual species occurrence record co-ordinate and date, this function extracts data for a given band within a Google Earth Engine dataset across a user-specified period and calculates a mathematical function on such data.

Value

Returns details of successful explanatory variable extractions.

Google Earth Engine

extract_dynamic_coords() requires users to have installed R package rgee (Aybar et al., 2020) and initialised Google Earth Engine with valid log-in credentials. Please follow instructions on the following website https://cran.r-project.org/package=rgee.

- datasetname must be in the accepted Google Earth Engine catalogue layout (e.g. "MODIS/006/MCD12Q1" "or "UCSB-CHG/CHIRPS/DAILY")
- bandname must be as specified under the dataset in the Google Earth Engine catalogue (e.g. "LC_Type5", "precipitation"). For datasets and band names, see https://developers.google.com/earth-engine/datasets.

Mathematical function

GEE.math.fun specifies the mathematical function to be calculated over the temporal period from each record's date. Options are limited to Google Earth Engine ImageCollection Reducer functions (https://developers.google.com/earth-engine/apidocs/) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

Categorical data

Please be aware, at current this function does not support the extraction of temporally dynamic variables for specific categories within categorical datasets.

When extracting from categorical datasets, be careful to choose appropriate mathematical functions for such data. For instance, "first" or "last" may be more relevant that "sum" of land cover classification numbers.

Exporting extracted data

For `save.method = combined`, the function with save "csv" files containing all occurrence records and associated values for the explanatory variable.

For `save.method = split`, the function will save individual "csv" files for each record with each unique period of the given temporal.level (e.g. each year, each year and month combination or each unique date).

`split` protects users if internet connection is lost when extracting data for large occurrence datasets. The argument `resume` can be used to resume to previous progress if connection is lost.

References

Aybar, C., Wu, Q., Bautista, L., Yali, R. and Barja, A., 2020. rgee: An R package for interacting with Google Earth Engine. Journal of Open Source Software, 5(51), p.2272.

Examples

```
data(sample_filt_data)



extract_dynamic_coords(occ.data=sample_filt_data,
 datasetname = "UCSB-CHG/CHIRPS/DAILY",
 bandname="precipitation",
 spatial.res.metres = 5566 ,
 GEE.math.fun = "sum",
 temporal.direction = "prior",
 temporal.res = 364,
 save.method = "split",
 resume = TRUE,
 varname = "total_annual_precipitation_prior",
 save.directory= tempdir())
```

---

extract_dynamic_raster

Extract temporally dynamic rasters of explanatory variables.

Description

Extract rasters for temporally dynamic explanatory variables at each projection date using Google Earth Engine.

Usage

```
extract_dynamic_raster(
  dates,
  spatial.ext,
  datasetname,
  bandname,
  spatial.res.metres,
  GEE.math.fun,
  user.email,
  varname,
  temporal.res,
  temporal.direction,
  save.directory,
  save.drive.folder,
  resume = TRUE
)
```

Arguments

dates	a character string, vector of dates in format "YYYY-MM-DD".
spatial.ext	the spatial extent for the extracted raster. Object from which extent can be extracted of class SpatExtent, SpatRaster, an sf polygon or a numeric vector listing xmin, xmax, ymin and ymax in order.
datasetname	a character string, the Google Earth Engine dataset to extract data from.
bandname	a character string, the Google Earth Engine dataset bandname to extract data for.
spatial.res.metres	a numeric value, specifying the spatial resolution in metres of the raster to be extracted.
GEE.math.fun	a character string, the mathematical function to compute across the specified time frame from each projection date and for each cell.
user.email	a character string, user email for initialising Google Drive.
varname	optional; a character string, the unique name for the explanatory variable. Default varname is “bandname_temporal.res_temporal.direction_ GEE.math.fun_raster".
temporal.res	a numeric value, the temporal resolution in days to extract data across.
temporal.direction	a character string, the temporal direction for extracting dynamic variable data across relative to each projection date given. One of prior or post: can be abbreviated.
save.directory	optional; a character string, path to local directory to save extracted rasters to.
save.drive.folder	optional; a character string, Google Drive folder name to save extracted rasters to. Folder must be uniquely named within your Google Drive. Do not provide path.
resume	a logical indicating whether to search save.directory or save.drive.folder and return to previous progress through projection dates.Default = TRUE.

Details

For each projection date, this function downloads rasters at a given spatial extent and resolution for temporally dynamic explanatory variables. For each cell within the spatial extent, the `GEE.math.fun` is calculated on the data extracted from across the specified number of days prior or post the projection date. Rasters can be saved to Google Drive or a local directory too. These rasters can be combined to create projection covariate data frames for projecting dynamic species distribution and abundance at high spatiotemporal resolution.

Value

Returns details of successful explanatory variable extractions for each projection date.

Google Earth Engine

`extract_dynamic_raster()` requires users to have installed the R package `rgee` (Aybar et al., 2020) and initialised Google Earth Engine with valid log-in credentials. Please follow instructions on the following website https://cran.r-project.org/package=rgee.

- `datasetname` must be in the accepted Google Earth Engine catalogue layout (e.g. "MODIS/006/MCD12Q1" or "UCSB-CHG/CHIRPS/DAILY")
- `bandname` must be as specified under the dataset in the Google Earth Engine catalogue (e.g. "LC_Type5", "precipitation"). For datasets and band names, see https://developers.google.com/earth-engine/datasets.

Google Drive

`extract_dynamic_raster()` also requires users to have installed the R package `googledrive`(D'Agostino McGowan and Bryan, 2022) and initialised Google Drive with valid log-in credentials, which must be stated using argument `user.email`. Please follow instructions on https://googledrive.tidyverse.org/ for initialising the `googledrive` package.

The `save.drive.folder` must be uniquely named within your Google Drive and do not provide the path.

Note: When running this function a folder labelled "dynamicSDM_download_bucket" will be created in your Google Drive. This will be emptied once the function has finished running and output rasters will be found in the `save.drive.folder` or `save.directory`.

Mathematical function

`GEE.math.fun` specifies the mathematical function to be calculated over the temporal period from each projection date. Options are limited to Google Earth Engine ImageCollection Reducer functions (https://developers.google.com/earth-engine/apidocs/) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

Categorical data

Please be aware, at current this function does not support the extraction of temporally dynamic variables for specific categories within categorical datasets.

When extracting from categorical datasets, be careful to choose appropriate mathematical functions for such data. For instance, "first" or "last" may be more relevant that "sum" of land cover classification numbers.

Occasional rgee errors

As this function uses the `rgee` package to extract rasters from Google Earth Engine, below we outline occasional `rgee` errors that may occur when extracting rasters:

- "To avoid memory excess problems, ee_as_raster will not build Raster objects for large images"

This can be a sporadic error. It may be related to GEE server usage or internet connection at the time you tested the function. Try restarting your R session or try again at another time. Also, try clearing the files from the "dynamicSDM_download_bucket" in your Google Drive.

This error could also be due to an issue with your input `spatial.res.metres`. This function will extract rasters at all typical spatial resolutions of remote-sensing data and at global extents. If this error persists, please ensure you have not accidentally given an unrealistically high spatial resolution (e.g. `spatial.res.metres = 0.01` when you may be confusing the spatial resolution in degrees with metres).

- "Pixel type not supported: Type Long. Convert the image to a floating point type or a smaller integer type, for example, using ee.Image.toDouble()"

This error appears when `rgee` has been given an input that cannot be extracted. Some common causes include:

- Inappropriate `GEE.math.fun` for extracting categorical data.
- Dates or spatial extents that are not covered by the chosen GEE dataset. Remember to check whether the first projection date minus `temporal.res` is still within the temporal extent of the dataset.

References

Aybar, C., Wu, Q., Bautista, L., Yali, R. and Barja, A., 2020. rgee: An R package for interacting with Google Earth Engine. Journal of Open Source Software, 5(51), p.2272.

D'Agostino McGowan L., and Bryan J., 2022. googledrive: An Interface to Google Drive. https://googledrive.tidyverse.org, https://github.com/tidyverse/googledrive.

Examples

```
dates <- dynamic_proj_dates("2018-01-01", "2018-12-01", interval = 3,interval.level = "month")


data("sample_extent_data")
user.email<-as.character(gargle::gargle_oauth_sitrep()$email)
```

```
extract_dynamic_raster(dates = dates,
                       datasetname = "UCSB-CHG/CHIRPS/DAILY",
                       bandname="precipitation",
                       user.email = user.email,
                       spatial.res.metres = 5566,
                       GEE.math.fun = "sum",
                       temporal.direction = "prior",
                       temporal.res = 56,
                       spatial.ext = sample_extent_data,
                       varname = "total_annual_precipitation_prior",
                       save.directory = tempdir())
```

`extract_static_coords` *Extract explanatory variables from static rasters*

Description

Explanatory variable data are extracted from static environmental rasters at record co-ordinate or across moving window matrix

Usage

```
extract_static_coords(
  occ.data,
  varnames,
  extraction.method = "simple",
  static.rasters,
  moving.window.matrix,
  GEE.math.fun
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", and associated explanatory variable data.
`varnames`	a character string or vector, the unique names for each explanatory variable in order of layers in the SpatRaster.
`extraction.method`	a character string or vector, the methods to extract data from SpatRaster using `terra` package `extract` function. One of `simple` or `bilinear`. If `simple` values for the cell a point falls in are returned. If `bilinear` the returned values are interpolated from the values of the four nearest raster cells.

`static.rasters` a `SpatRaster` containing one or more SpatRaster layers to extract data from.

`moving.window.matrix`
optional; a matrix of weights with an odd number of sides, representing the spatial neighbourhood of cells ("moving window") to calculate `GEE.math.fun` across from record co-ordinate. See details for more information.

`GEE.math.fun` optional; a character string, the mathematical function to compute across the specified spatial matrix for each record.

Details

Function to extract data from static rasters either at occurrence record co-ordinates or spatially buffered using a moving window matrix.

Note:

- `varnames` must be in the order of raster layers within the SpatRaster.
- `extraction.method` must be of length one to apply to all layers, or length equal to the number of layers in `static.rasters`.

Value

Returns the occurrence data frame with added columns for extracted data.

Spatial buffering (optional)

Using the `focal` function from `terra` R package (Hijmans et al., 2022), `GEE.math.fun` is calculated across the spatial buffer area from the record co-ordinate. The spatial buffer area used is specified by the argument `moving.window.matrix`, which dictates the neighbourhood of cells surrounding the cell containing the occurrence record to include in this calculation.

See function `get_moving_window()` to generate appropriate `moving.window.matrix`.

Mathematical function

`GEE.math.fun` specifies the mathematical function to be calculated over the spatial buffered area and temporal period. Options are limited to Google Earth Engine ImageCollection Reducer functions (https://developers.google.com/earth-engine/apidocs/) for which an analogous R function is available. This includes: "allNonZero","anyNonZero", "count", "first","firstNonNull", "last", "lastNonNull", "max","mean", "median","min", "mode","product", "sampleStdDev", "sampleVariance", "stdDev", "sum" and "variance".

References

Hijmans, R.J., Bivand, R., Forner, K., Ooms, J., Pebesma, E. and Sumner, M.D., 2022. Package 'terra'. Maintainer: Vienna, Austria.

Examples

```
data("sample_explan_data")
random_cat_layer <- terra::rast(sample_extent_data)
random_cat_layer <- terra::setValues(random_cat_layer,
                                     sample(0:10, terra::ncell(random_cat_layer),
                                            replace = TRUE))

extract_static_coords(occ.data = sample_explan_data,
                      varnames = "random_cat_layer",
                      static.rasters = random_cat_layer)
```

---

get_moving_window *Generate a "moving window" matrix of optimal size*

Description

Calculates an optimal "moving window" matrix size for use when extracting spatially buffered explanatory variables, by using the radius of interest and spatial resolution of environmental data.

Usage

```
get_moving_window(
  radial.distance,
  spatial.res.degrees,
  spatial.res.metres,
  spatial.ext
)
```

Arguments

`radial.distance`	a numeric value, the radius of interest in metres.
`spatial.res.degrees`	a numeric value, the spatial resolution in degrees of explanatory variable data.
`spatial.res.metres`	a numeric value, the spatial resolution in metres of explanatory variable data.
`spatial.ext`	the spatial extent of the study. Object from which extent can be extracted of class `SpatExtent`, `SpatRaster`, `sf` polygon or numeric vector listing xmin, xmax, ymin and ymax in order.

Value

Returns "moving window" matrix with an odd number of sides and equal weights.

Importance for other functions in `dynamicSDM` **To extract spatially buffered explanatory**

variable data using dynamicSDM functions `extract_buffered_coords()` or `extract_buffered_raster()`, a "moving window" matrix specifying the neighbourhood of cells to include in the calculation is required.

For example, by using a three by three "moving window" matrix of equal weights, the explanatory variable would be calculated across the nine grid cells neighbouring the cell of interest and the cell of interest.

Why use a moving window matrix instead of circular buffer?

The benefit of using a "moving window" over calculating explanatory variable values across a set radius from each record co-ordinate, is that when generating projection rasters at high spatial and temporal resolution, these can be generated much faster as the "moving windows" standardise the calculation.

Function calculation

- 1. To calculate the "moving window" matrix size, the `get_moving_window()` function first calculates the circular area of interest, using the user-specified radius of interest and the equation for area of a circle.

This radius should be chosen to represent the radial distance from species occurrence record co-ordinates that the explanatory variable data might be relevant and impact species presence.

- 1. Then, the average grid cell area of the explanatory variable data (derived from user-provided spatial resolution and extent) is calculated. If `spatial.res.degrees` is given then `spatial.ext` is required to calculate average cell area size. If `spatial.res.metres` is given then average cell area is calculated by squaring this value to get cell area in square metres.
- 1. Finally, the function then calculates the optimal "moving window" matrix that best matches circular area of interest with the "moving window" matrix area. The matrix of weights will have an odd number of sides.

Examples

```
get_moving_window(radial.distance = 100000, spatial.res.metres = 111320)
```

`sample_cov_data` *Sample projection covariates three variables across for southern Africa.*

Description

Data frame of co-ordinates and associated dynamic explanatory variable values for "2018-04-01" cropped to southern Africa at 2 degree resolution.

Usage

```
sample_cov_data
```

Format

A data frame with 225 rows and 6 variables

X row name

x grid cell longitude

y grid cell latitude

eight_sum_prec sum Climate Hazards Group InfraRed Precipitation With Station Data (Funk et al., 2015) total daily precipitation at record co-ordinate across 52-weeks prior to "2018-04-01" (mm).

grass_crop_percentage total number of MODIS Land Cover Type Yearly 500m (Friedl et al., 2019) "cereal cropland" and "grassland" cells in surrounding area of record co-ordinate in 2018.

year_sum_prec sum Climate Hazards Group InfraRed Precipitation With Station Data (Funk et al., 2015) total daily precipitation at record co-ordinate across 52-weeks prior to "2018-04-01" (mm).

References

Friedl, M., Sulla-Menashe, D. (2019). MCD12Q1 MODIS/Terra+Aqua Land Cover Type Yearly L3 Global 500m SIN Grid V006. NASA EOSDIS Land Processes DAAC. Accessed 2022-11-24 from

Funk, Chris, Pete Peterson, Martin Landsfeld, Diego Pedreros, James Verdin, Shraddhanand Shukla, Gregory Husak, James Rowland, Laura Harrison, Andrew Hoell & Joel Michaelsen. "The climate hazards infrared precipitation with stations-a new environmental record for monitoring extremes". Scientific Data 2, 150066.

---

sample_events_data *Sample e-Bird sampling event records*

Description

A dataset containing a sample of e-Bird sampling events for all bird species across southern Africa between 2000-2020 (Fink et al., 2021, GBIF, 2021). The variables are as follows:

Usage

```
sample_events_data
```

Format

A data frame with 1000 rows and 5 variables:

day avian e-Bird sampling event day.

month avian e-Bird sampling event month.

year avian e-Bird sampling event year.

y avian e-Bird sampling event latitude.

x avian e-Bird sampling event longitude.

References

Fink, D., T. Auer, A. Johnston, M. Strimas-Mackey, O. Robinson, S. Ligocki, W. Hochachka, L. Jaromczyk, C. Wood, I. Davies, M. Iliff, L. Seitz. 2021. eBird Status and Trends, Data Version: 2020; Released: 2021. Cornell Lab of Ornithology, Ithaca, New York. doi: 10.2173/ebirdst.2020

GBIF.org (12 July 2021) GBIF Occurrence Download doi: 10.15468/dl.ppcu6q

---

`sample_explan_data` *Sample species occurrence records with associated dynamic explanatory variables*

Description

A dataset containing a sample of the bird species, the red-billed quelea (Quelea quelea), distribution records from between 2002-2019 (GBIF 2021, GBIF 2022); generated pseudo-absence records, and associated extracted dynamic explanatory variables. The variables are as follows:

Usage

```
sample_explan_data
```

Format

A data frame with 330 rows and 17 variables:

x species occurrence record longitude.

y species occurrence record latitude.

year species occurrence record year.

month species occurrence record month.

day species occurrence record day.

decimalLatitude species occurrence record latitude.

decimalLongitude species occurrence record longitude.

occurrenceStatus species presence or absence character.

source source of occurrence or pseudo-absence data point.

species name of species occurrence records belong to name

SAMP_EFFORT total number of avian e-Bird sampling events within spatiotemporal buffer of occurrence record location and dates.

REL_SAMP_EFFORT proportion of total number of avian e-Bird sampling events within spatiotemporal buffer of occurrence record location and dates relative to other records

unique.ID.DYN unique id value assigned when extracting dynamic explanatory variable data

eight_sum_prec sum Climate Hazards Group InfraRed Precipitation With Station Data (Funk et al., 2016) total daily precipitation at record co-ordinate across 52-weeks prior to record date (mm).

grass_crop_percentage total number of MODIS Land Cover Type Yearly 500m (Friedl & Sulla-Menashe, 2019) "cereal cropland" and "grassland" cells in surrounding area of record co-ordinate in record year.

year_sum_prec sum Climate Hazards Group InfraRed Precipitation With Station Data (CHIRPS Daily) total daily precipitation at record co-ordinate across 52-weeks prior to record date (mm).

presence.absence binary species presence or absence at record location and date.

References

Friedl, M., Sulla-Menashe, D. (2019). MCD12Q1 MODIS/Terra+Aqua Land Cover Type Yearly L3 Global 500m SIN Grid V006. NASA EOSDIS Land Processes DAAC.

Funk, Chris, Pete Peterson, Martin Landsfeld, Diego Pedreros, James Verdin, Shraddhanand Shukla, Gregory Husak, James Rowland, Laura Harrison, Andrew Hoell & Joel Michaelsen. "The climate hazards infrared precipitation with stations-a new environmental record for monitoring extremes". Scientific Data 2, 150066. doi:10.1038/sdata.2015.66 2015.

GBIF.org (12 July 2021) GBIF Occurrence Download doi: 10.15468/dl.ppcu6q

GBIF.org (25 July 2022) GBIF Occurrence Download doi: 10.15468/dl.k2kftv

---

sample_extent_data *MULTIPOLYGON object for the extent of southern Africa*

Description

A MULTIPOLYGON (package "sf") object containing polygons for each country within southern Africa. The variables are as follows:

Usage

```
sample_extent_data
```

Format

A simple feature collection with 10 features and 1 field.

geometry MULTIPOLYGON object co-ordinates for country boundaries.

name name of country the polygon represents.

`sample_filt_data` *Sample of filtered species occurrence records*

Description

A dataset containing a sample of the bird species, the red-billed quelea (Quelea quelea), distribution records (GBIF 2021 & GBIF 2022) that have been filtered to special extent of southern Africa and quality checked using `dynamicSDM` functions. The variables are as follows:

Usage

```
sample_filt_data
```

Format

A data frame with 330 rows and 12 variables:

x species occurrence record x

y species occurrence record y

year species occurrence record year.

month species occurrence record month.

day species occurrence record day.

decimalLatitude species occurrence record latitude.

decimalLongitude species occurrence record longitude.

occurrenceStatus species presence or absence character.

source source of occurrence or pseudo-absence data point.

species name of species occurrence records belong to name

SAMP_EFFORT total number of avian e-Bird sampling events within spatiotemporal buffer of occurrence record location and dates.

REL_SAMP_EFFORT proportion of total number of avian e-Bird sampling events within spatiotemporal buffer of occurrence record location and dates relative to other records

References

GBIF.org (12 July 2021) GBIF Occurrence Download doi: 10.15468/dl.ppcu6q

GBIF.org (25 July 2022) GBIF Occurrence Download doi: 10.15468/dl.k2kftv

sample_occ_data *Sample species occurrence records*

Description

A dataset containing a sample of the bird species, the red-billed quelea (Quelea quelea), distribution records between 1976-2021 (GBIF 2021 & GBIF 2022). The variables are as follows:

Usage

```
sample_occ_data
```

Format

A data frame with 600 rows and 7 variables:

year species occurrence record year.

month species occurrence record month.

day species occurrence record day.

decimalLatitude species occurrence record latitude.

decimalLongitude species occurrence record longitude.

occurrenceStatus species presence or absence character.

source source of occurrence or pseudo-absence data point.

References

GBIF.org (12 July 2021) GBIF Occurrence Download doi: 10.15468/dl.ppcu6q

GBIF.org (25 July 2022) GBIF Occurrence Download doi: 10.15468/dl.k2kftv

spatiotemp_autocorr *Test for spatial and temporal autocorrelation in species distribution model explanatory data.*

Description

Function performs statistical tests to assess spatial and temporal autocorrelation in given explanatory variable data.

Usage

```
spatiotemp_autocorr(
  occ.data,
  varname,
  temporal.level,
  plot = FALSE,
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

occ.data	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day" and associated explanatory data.
varname	a character string or vector, the name(s) of the columns within occ.data containing data to test for autocorrelation.
temporal.level	a character string or vector, the time step(s) to test for temporal autocorrelation at. One or multiple of day or month, year. Can be abbreviated.
plot	a logical indicating whether to generate plot of temporal autocorrelation. See details for plot description. Default = FALSE.
prj	a character string, the coordinate reference system of occurrence data. Default is "+proj=longlat +datum=WGS84".

Details

To test for temporal autocorrelation, the function first calculates the average value across records for each time step (temporal.level). The correlation between the average value at one time point (t) and the value at the previous time point (t-1) is calculated and plotted (if plot = TRUE) A significant relationship between values at consecutive data points indicates temporal autocorrelation is present.

To test for spatial autocorrelation, the function calculates a distance matrix between all record co-ordinates. Moran's I statistical test is calculated to test whether points closer in space have more similar values than those more distant from each other(Legendre, 1993). Please note that NA values are removed before Moran's I calculation.

As the spatial autocorrelation calculation involves computation of a distance matrix between all occurrence records. To reduce computation time, it is recommended that a sample of large occurrence datasets are input.

Value

Returns a list of temporal and spatial autocorrelation test results for each variable.

References

Legendre, P. J. E. 1993. Spatial Autocorrelation: Trouble Or New Paradigm? 74, 1659-1673.

Examples

```
data("sample_explan_data")
spatiotemp_autocorr(sample_explan_data,
                    varname = c("year_sum_prec","eight_sum_prec"),
                    temporal.level = c("year","month","day"))
```

`spatiotemp_bias` *Test for spatial and temporal bias in species occurrence records*

Description

Generates plots for visual assessment of spatial and temporal biases in occurrence records. Tests whether the spatiotemporal distribution of records is significantly different from the distribution from random sampling.

Usage

```
spatiotemp_bias(
  occ.data,
  temporal.level,
  plot = FALSE,
  spatial.method = "simple",
  centroid,
  radius,
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
`temporal.level`	a character string or vector, the time step(s) to test for temporal bias at. One or multiple of `day` or `month`, `year`. Can be abbreviated.
`plot`	a logical indicating whether to generate plots of spatial and temporal bias. See details for plot descriptions. Default = `FALSE`.
`spatial.method`	a character string, the method to calculate the spatial bias statistic. One of; `simple`, `convex_hull` or `core`. See details.
`centroid`	a numeric vector of length two, specifying the centroid co-ordinates in the order of longitude then latitude. Only required if `spatial.method = core`. Default is mean of all occurrence record co-ordinates.
`radius`	a numeric value, the radial distance in metres from the given centroid co-ordinate to measure spatial bias within. Only required if `spatial.method = core`. See details for more information. Default is mean distance of all co-ordinates from `centroid`.
`prj`	a character string, the coordinate reference system of occ.data co-ordinates. Default is "+proj=longlat +datum=WGS84".

Value

Returns list containing chi-squared and t-test results, and plots if specified.

Temporal bias

To assess temporal sampling bias, the function returns a histogram plot of the frequency distribution of records across the given time step specified by `temporal.level` (if `plot = TRUE`). The observed frequency of sampling across the categorical time steps are compared to the distribution expected from random sampling, using a chi-squared test (Greenwood and Nikulin, 1996) .

Spatial bias

To assess spatial sampling bias, the function returns a scatter plot of the spatial distribution of occurrence records to illustrate any spatial clustering (if `plot = TRUE`). The average nearest neighbour distance of record co-ordinates is then compared to that of records randomly generated at same density using a t-test, following the nearest neighbour index established by Clark and Evans (1954).

Bias: methods

Below we outline the methods for which these tests for biases can be applied. `dynamicSDM` offers the additional functionality of the `core` approach. This enables users to explore sampling biases in set areas of a species range. This may be valuable if periphery-core relationships could lead to inaccurate inferences of sampling bias. For instance, if species are expanding or shifting their ranges through space and time. #'

- `simple` - generates the random points within a rectangle created using the minimum and maximum longitude and latitude of occurrence co-ordinates.
- `convex_hull` - generates the random points within the convex hull of occurrence record co-ordinates (i.e. the smallest convex set that contains all records).
- `core` - generates the random points within specified circular area generated from a centroid point and radius. If these arguments (`centroid` and `radius`) are not provided then `centroid` is calculated by averaging co-ordinates of all occurrence records, and `radius` is the mean distance away of all records from the centroid.

For each method, only occurrence records within the specified area are tested for spatial and temporal sampling biases.

Computation time

As the spatial bias test involves the calculation of a distance matrix. To reduce computation time, it is recommended that only a representative sample of large occurrence datasets are input.

References

Clark, P. J. & Evans, F. C. J. E. 1954. Distance To Nearest Neighbor As A Measure Of Spatial Relationships In Populations. 35, 445-453.

Greenwood, P. E. & Nikulin, M. S. 1996. A Guide To Chi-Squared Testing, John Wiley & Sons.

Examples

```
data(sample_explan_data)
```

```
bias_simple <- spatiotemp_bias(
occ.data = sample_explan_data,
temporal.level = c("year"),
spatial.method = "simple",
plot = FALSE
)
```

`spatiotemp_block` *Split occurrence records into spatial and temporal blocks for model fitting.*

Description

Splits occurrence records into spatial and temporal sampling units and groups sampling units into multiple blocks that have similar mean and range of environmental explanatory variables and sample size.

Usage

```
spatiotemp_block(
  occ.data,
  vars.to.block.by,
  spatial.layer,
  spatial.split.degrees,
  temporal.block,
  n.blocks = 10,
  iterations = 5000
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day", and associated explanatory variable data.
`vars.to.block.by`	a character string or vector, the explanatory variable column names to group sampling units based upon.
`spatial.layer`	optional; a `SpatRaster` object, a categorical spatial layer for sample unit splitting.
`spatial.split.degrees`	a numeric value, the grid cell resolution in degrees to split `spatial.layer` by. Required if `spatial.layer` given.
`temporal.block`	optional; a character string or vector, the time step for sampling unit splitting. Any combination of `day`, `month`, `year` or `quarter`. See details.
`n.blocks`	optional; a numeric value of two or more, the number of blocks to group occurrence records into. Default; 10.

`iterations` optional; a numeric value, the number of random block groupings to trial before selecting the optimal grouping. Default; 5000.

Value

Returns occurrence data frame with column "BLOCK.CATS", assigning each record to a spatiotemporal block.

Blocking for autocorrelation

Blocking is an established method to account for spatial autocorrelation in SDMs. Following Bagchi et al., (2013), the blocking method involves splitting occurrence data into sampling units based upon non-contiguous ecoregions, which are then grouped into spatially disaggregated blocks of approximately equal sample size, within which the mean and range of explanatory variable data are similar. When species distribution model fitting, blocks are left out in-turn in a jack-knife approach for model training and testing.

We adapt this approach to account for temporal autocorrelation by enabling users to split records into sampling units based upon spatial and temporal characteristic before blocking occurs.

Spatial splitting

If the `spatial.layer` has categories that take up large contiguous areas, `spatiotemp_block()` will split categories into smaller units using grid cells at specified resolution (`spatial.split.degrees`).

Temporal splitting

If `temporal.block` is given, then occurrence records with unique values for the given level are considered unique sampling unit. For instance, if `temporal.block = year`, then records from the same year are considered a sampling unit to be grouped into blocks.

Note: If spatial splitting is also used, then spatial characteristics may split these further into separate sampling units.

The `temporal.block` option `quarter` splits occurrence records into sampling units based on which quarter of the year the record month belongs to: (1) January-March, (2) April-June, (3) July-September and (4) October-December. This could be employed if seasonal biases in occurrence record collection are driving autocorrelation.

Block generation

Once split into sampling units based upon temporal and spatial characteristics, these units are then assigned into given number of blocks (`n.blocks`), so that the mean and range of explanatory variables (`vars.to.block.by`) and total sample size are similar across each. The number of `iterations` specifies how many random shuffles are used to optimise block equalisation.

References

Bagchi, R., Crosby, M., Huntley, B., Hole, D. G., Butchart, S. H. M., Collingham, Y., Kalra, M., Rajkumar, J., Rahmani, A. & Pandey, M. 2013. Evaluating the effectiveness of conservation site networks under climate change: accounting for uncertainty. Global Change Biology, 19, 1236-1248.

Examples

```
data("sample_explan_data")
data("sample_extent_data")
random_cat_layer <- terra::rast(sample_extent_data)
random_cat_layer <- terra::setValues(random_cat_layer,
                                     sample(0:10, terra::ncell(random_cat_layer),
                                            replace = TRUE))

spatiotemp_block(
 occ.data = sample_explan_data,
 spatial.layer = random_cat_layer,
 spatial.split.degrees = 3,
 temporal.block = c("month"),
 vars.to.block.by = colnames(sample_explan_data)[14:16],
 n.blocks = 3,
 iterations = 30
)
```

---

spatiotemp_check *Check species occurrence record formatting, completeness and validity.*

Description

Checks the occurrence record data frame contains the column names and classes required for dynamicSDM functions. Option to exclude records containing missing, duplicate or invalid coordinates or dates.

Usage

```
spatiotemp_check(
  occ.data,
  na.handle,
  duplicate.handle,
  coord.handle,
  date.handle,
  date.res,
  coordclean = FALSE,
  coordclean.species,
  coordclean.handle = "exclude",
  ...
)
```

Arguments

occ.data	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
na.handle	a character string, method for handling missing data (NA values) in record co-ordinates and dates. One of exclude or ignore: can be abbreviated. Default; exclude.
duplicate.handle	a character string, method for handling duplicate record co-ordinates or dates. One of exclude or ignore: can be abbreviated. Default; exclude.
coord.handle	a character string, method for handling invalid co-ordinates in record data. One of exclude or ignore: can be abbreviated. Default; exclude.
date.handle	a character string, method for handling invalid dates in record data. One of exclude or ignore: can be abbreviated. Default; exclude.
date.res	a character string, stating the temporal resolution to complete checks on. One of year, month or day. If not given, dates are not checked.
coordclean	a logical indicating whether to run function clean_coordinates from package CoordinateCleaner on occ.data. Default = FALSE.
coordclean.species	a character string or vector, specifying the name of the species that all of occ.data records belong to, or a character vector the length of nrow(occ.data) specifying which species each record belongs to. Required if coordclean = TRUE.
coordclean.handle	a character string, method for handling records flagged by CoordinateCleaner. One of exclude or report. Default: exclude.
...	Other arguments passed onto CoordinateCleaner.

Value

By default, returns occurrence record data frame, filtered to exclude records containing missing, duplicate or invalid data in record co-ordinates and dates.

date.res argument

The date.res states the temporal resolution to check dates, including when searching for duplicate records, removing records with NA values and checking for invalid dates.

Validity checks

Record dates and co-ordinates are checked for validity using the following rules:

- Dates must be real dates that could exist. For example, 50th February 2000 is not a valid date.
- Co-ordinates must have longitude (x) values between -180 and 180, and latitude (y) values between -90 and 90 to be considered valid.

CoordinateCleaner compatibility

spatiotemp_check() acts as a helper function for compatibility with the R package CoordinateCleaner (Zizka et al., 2019), which offers a diversity of functions for checking the co-ordinates of occurrence records.

If coordclean = TRUE, then coordclean.species must be provided to identify which species each record belonds to. If coordclean.handle = exclude then all occ.data records flagged by CoordinateCleaner::clean_coordinates() as potentially erroneous are removed in the returned data.

If coordclean.handle = report, then the occurrence data frame is returned with an additional CC_REPORT column. This column contains the output from CoordinateCleaner::clean_coordinates() which indicates the potentially erroneous records.

References

Zizka A, Silvestro D, Andermann T, Azevedo J, Duarte Ritter C, Edler D, Farooq H, Herdean A, Ariza M, Scharn R, Svanteson S, Wengstrom N, Zizka V, Antonelli A (2019). "CoordinateCleaner: standardized cleaning of occurrence records from biological collection databases." Methods in Ecology and Evolution, -7. doi: 10.1111/2041210X.13152, R package version 2.0-20, https://github.com/ropensci/CoordinateCleaner.

Examples

```
data(sample_occ_data)
sample_occ_data<-convert_gbif(sample_occ_data)

nrow(sample_occ_data)

filtered<-spatiotemp_check(
 occ.data = sample_occ_data,
 coord.handle = "exclude",
 date.handle = "exclude",
 duplicate.handle = "exclude",
 na.handle = "exclude"
)
nrow(filtered)


filtered_CC<-spatiotemp_check(
 occ.data = sample_occ_data,
 coord.handle = "exclude",
 date.handle = "exclude",
 duplicate.handle = "exclude",
 na.handle = "exclude",
 coordclean = TRUE,
 coordclean.species = "quelea",
 coordclean.handle = "exclude"
)
nrow(filtered_CC)
```

`spatiotemp_extent` *Filter species occurrence records by a given spatial and temporal extent.*

Description

Function excludes species occurrence records with co-ordinates outside a given spatial extent and record dates outside a given temporal extent.

Usage

```
spatiotemp_extent(
  occ.data,
  temporal.ext,
  spatial.ext,
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
`temporal.ext`	optional; a character vector, two dates in format "YYYY-MM-DD". First date represents start of temporal extent and second date represents end of temporal extent for inclusion.
`spatial.ext`	the spatial extent to filter by. Object from which extent can be extracted of class `SpatExtent`, `SpatRaster`, `sf` polygon or numeric vector listing xmin, xmax, ymin and ymax in order.
`prj`	a character string, the coordinate reference system of input `occ.data` co-ordinates. Default is "+proj=longlat +datum=WGS84".

Value

Returns data frame of occurrence records filtered to the spatial and temporal extent given.

Spatial extent

If `spatial.ext` is provided, `spatiotemp_extent()` checks whether species occurrence record co-ordinates are within the given spatial extent of the study (`spatial.ext`) and excludes any outside of this extent.

If `spatial.ext` object can be used as a mask by `terra::mask()` then the mask is used to filter records in a more targeted way. If not, then the rectangular extent of the `spatial.ext` object is used.

Temporal extent

If `temporal.ext` is provided, `spatiotemp_extent()` checks whether species occurrence record dates are within the given temporal extent of the study and excludes any outside of this extent.

Examples

```
data(sample_filt_data)
data(sample_extent_data)

results <- spatiotemp_extent(occ.data = sample_filt_data,
                             spatial.ext = sample_extent_data,
                             temporal.ext = c("2012-01-01", "2017-01-01"))
```

`spatiotemp_pseudoabs` *Generate pseudo-absence record coordinates and dates*

Description

Function generates specified number of pseudo-absence record co-ordinates and dates either randomly or buffered in space and time.

Usage

```
spatiotemp_pseudoabs(
  spatial.method,
  temporal.method,
  occ.data,
  spatial.ext,
  temporal.ext,
  spatial.buffer,
  temporal.buffer,
  n.pseudoabs = 100,
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

`spatial.method`	a character string, the spatial method for pseudo-absence generation. One of `buffer` or `random`: can be abbreviated.
`temporal.method`	a character string, the temporal method for pseudo-absence generation. One of `buffer` or `random`: can be abbreviated.
`occ.data`	optional; a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day". Required if either `temporal.method` or `spatial.method` is `buffer`.

spatial.ext	the spatial extent to randomly generate pseudo-absences within. Object from which extent can be extracted of class SpatExtent, SpatRaster, an sf polygon or a numeric vector listing xmin, xmax, ymin and ymax in order. Required if spatial.method is random, and optionally used if buffer. See details.
temporal.ext	optional; a character vector, two dates in format "YYYY-MM-DD". The first represents the start of the temporal extent and the second represents the end of temporal extent to randomly generate pseudo-absences dates within. Required if temporal.method is random, and optionally used if buffer. See details.
spatial.buffer	optional; a numeric value or vector, the radius/radii in metres to generate buffered pseudo-absence coordinates within. Only required if spatial.method is buffer. See details.
temporal.buffer	optional; a numeric value or vector, the period(s) in days to generate buffered pseudo-absence dates within. Only required if temporal.method is buffer . See details.
n.pseudoabs	optional; a numeric value, the number of pseudo-absence records to generate. Default; 100.
prj	a character string, the coordinate reference system of input occ.data co-ordinates. Default is "+proj=longlat +datum=WGS84".

Details

Below we outline the various approaches to generating pseudo-absences through space and time available in the dynamicSDM package. To select the appropriate pseudo-absence generation approach and buffer size, there are many considerations. We recommend seeking the appropritae literature to inform your decision when species distribution modelling (Barbet-Massin et al., 2012, Phillips et al., 2009, Vanderwal et al., 2009).

Value

Returns data frame of pseudo-absence coordinates and dates.

Spatial buffer

If spatial.method is buffer, then the pseudo-absence record co-ordinates are randomly generated in a buffered area defined either by

- single numeric value for spatial.buffer - anywhere between the occurrence record and the circular distance surrounding this point (as specified in metres).
- two numeric values for spatial.buffer - anywhere between the closest radius from the occurrence record and the furthest away radius (as specified in metres).

For example, if spatial.buffer = c(3000,10000), then pseudo-absence co-ordinates are randomly generated at least 3000m radius away from occurrence record co-ordinate but within 10000m radius. Whereas, if spatial.buffer = 10000, then pseudo-absence co-ordinates are randomly generated anywhere between 0m and 10000m radius from the occurrence record.

If spatial.ext is given too, then the generated pseudo-absences are not only constrained to the buffered area but also to this extent. For instance, if occurrence records are coastal, you may want to clip buffers to only terrestrial regions using a country polygon given in spatial.ext.

Spatial random

If `spatial.method` is `random`, then the pseudo-absence record co-ordinates are randomly generated across `spatial.ext` object given.

If `spatial.ext` is a `sf` polygon or `SpatRaster` (mask if possible before input) then these shapes are used, instead of a simple rectangular extent (`SpatExtent`). Therefore, inputting one of these objects will allow for more specific pseudo-absence generation.

For example, inputting an `sf` polygon of a specific countries will ensure co-ordinates are terrestrial, whereas an extent (xmin, xmax, ymin, ymax) that encompasses these countries may result in the generation of pseudo-absence records in inappropriate areas, such as oceans or non-study-area countries.

Temporal buffer

If `temporal.method` is `buffer`, then pseudo-absence record dates are randomly generated between in a period defined by:

- single numeric value for temporal.buffer - any date between the occurrence record date and the total number of days specified prior or post.
- two numeric values for temporal.buffer - any date between the closest and furthers away number of days specified.

For example, if `temporal.buffer = c(14,30)`, then pseudo-absence dates randomly generated at least 14 days from occurrence record dates but within 30 days. Whereas if `temporal.buffer = 30`, pseudo-absence dates are randomly generated anywhere between 0 and 30 days prior or post the occurrence record date.

If `temporal.ext` is given too, then the generated pseudo-absence dates are not only constrained to the buffer period but also to this temporal extent. For instance, an occurrence record recorded at the start of `temporal.ext` with 7 day buffer, may result in generated pseudo-absences outside of the temporal extent of the study.

Temporal random

If `temporal.method` is `random`, then pseudo-absence record dates are randomly generated within the two `temporal.ext` dates given.

References

Barbet-Massin, M., Jiguet, F., Albert, C. H., Thuiller, W. J. M. I. E. & Evolution 2012. Selecting Pseudo-Absences For Species Distribution Models: How, Where And How Many? 3, 327-338.

Phillips, S. J., Dudik, M., Elith, J., Graham, C. H., Lehmann, A., Leathwick, J. & Ferrier, S. 2009. Sample Selection Bias And Presence-Only Distribution Models: Implications For Background And Pseudo-Absence Data. 19, 181-197.

Vanderwal, J., Shoo, L. P., Graham, C. & Williams, S. E. 2009. Selecting Pseudo-Absence Data For Presence-Only Distribution Modeling: How Far Should You Stray From What You Know? Ecological Modelling, 220, 589-594.

Examples

```
data("sample_filt_data")


spatiotemp_pseudoabs(
 sample_filt_data,
 spatial.method = "random",
 temporal.method = "random",
 spatial.ext = c(20, 36, -35, -12),
 temporal.ext = c("2011-01-01", "2017-01-01")
)
```

`spatiotemp_resolution` *Filter species occurrence records by given spatial and temporal resolution*

Description

Filters species occurrence record data frame to exclude records with co-ordinates and dates that do not meet specified spatial and temporal resolution.

Usage

```
spatiotemp_resolution(occ.data, spatial.res, temporal.res)
```

Arguments

`occ.data`	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
`spatial.res`	optional; a numeric value, the minimum acceptable number of decimal places given for occurrence record co-ordinates.
`temporal.res`	optional; a character string, the minimum acceptable temporal resolution of occurrence record dates. One of `day` , `month` or `year`: can be abbreviated.

Details

Excludes species occurrence records that do not meet the minimum spatial and temporal resolution specified.

If `spatial.res` given, the value of 1 represents an acceptable co-ordinate resolution of one decimal place, roughly equal to 11.1km, and value of 3 represents three decimal places, roughly equal to 111m.

If `temporal.res` given, `temporal.res` = `day` would result in exclusion of records without values for year, month and day, and `temporal.res` = `year` would only exclude records without values for year.

spatial.res and temporal.res can be informed based upon the highest spatial and temporal resolution of the datasets to be utilised when extracting dynamic variables.

For example, if explanatory variables datasets are annual, then a temporal.res of year is adequate, whereas if datasets are daily, then temporal.res of day may be more appropriate.

Value

Returns a data frame of species records filtered by the minimum acceptable spatial resolution of co-ordinates and temporal resolution of dates.

Examples

```
data(sample_occ_data)
sample_occ_data <- convert_gbif(sample_occ_data)

spatial_res_high <- spatiotemp_resolution(sample_occ_data, spatial.res = 4)

spatial_res_low <- spatiotemp_resolution(sample_occ_data, spatial.res = 1)

temporal_res <- spatiotemp_resolution(sample_occ_data, temporal.res = "day")
```

---

spatiotemp_thin *Thin species occurrence records by spatial and temporal proximity.*

Description

Thins species occurrence records that are within minimum spatial and temporal distance apart.

Usage

```
spatiotemp_thin(
  occ.data,
  temporal.method,
  temporal.dist,
  spatial.split.degrees,
  spatial.dist = 0,
  iterations = 100,
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

occ.data	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
temporal.method	a character string, the method to calculate temporal distance between records. One of DOY or day. See details for more information.

`temporal.dist` a numeric value, the temporal buffer in days to thin records by.

`spatial.split.degrees`
a numeric value, the grid cell resolution in degrees to split occurrence records by before temporal thinning.

`spatial.dist` a numeric value, the spatial buffer distances in metres to thin records by. Default no spatial thinning.

`iterations` a numeric value, the number of iterations to randomly thin occurrence records by. Default; 100.

`prj` a character string, the coordinate reference system of occ.data co-ordinates. Default is "+proj=longlat +datum=WGS84".

Value

Returns data frame of occurrence records thinned by specified temporal and spatial distance.

Overview

`spatiotemp_thin()` calculates the temporal distance between occurrence records in given area and excludes records below minimum temporal distance apart. Then calculates the spatial distance between all occurrence records and filters records below the minimum spatial distance apart using the `spThin` package function for spatial thinning (Aiello-Lammens et al., 2015). This approach has been shown to improve species distribution model performance (Boria et al., 2014).

Temporal thinning methods

For temporal thinning, the function first splits occurrence records into grid cells of given size in degrees (set by `spatial.split.degrees`). This is to prevent spatially distant but temporally close records from being excluded. For each grid cell, all records within the cell are temporally thinned. This process works by removing records that are within given temporal distance (`temporal.dist`) from each other by randomly selecting one of the two. This iterates through until no records are within the given temporal distance of each other in each grid cell, following a similar algorithm to `spThin` (Aiello-Lammens et al., 2015).

Two methods exist for measuring the temporal distance between occurrence records.

- 1. `doy` - calculates the minimum days apart within the annual cycle
- 1. `day` - uses the absolute number of days.

For instance, two dates “2010-01-05” and “2012-12-05” can be calculated as either 1065 absolute days apart, or within the annual cycle these dates represent day 5 and day 339 of the year, and are 31 days apart. Therefore, thinning by 40 days using the `DOY` method would remove one of these records, but using the `day` method would not. The chosen `temporal.method` will depend upon whether bias towards a point within the annual cycle or a point in linear time.

Spatial thinning

Following temporal thinning, spatial thinning occurs across entire dataset. The spatial distance between each record is calculated, and records within the given spatial distance (`spatial.dist`) from each other are excluded by randomly selecting one of these. This iterates through until no

records are with the given spatial distances of each other across entire dataset using the package `spThin` (Aiello-Lammens et al., 2015).

As random selection could alter the total number of occurrence records remaining in the occurrence record dataset, this process is iterated through a specified number of times (`iterations`) and the thinned data frame with the highest number of records remaining is returned.

References

Aiello-Lammens, M. E., Boria, R. A., Radosavljevic, A., Vilela, B. & Anderson, R. P. 2015. spThin: an R package for spatial thinning of species occurrence records for use in ecological niche models. Ecography, 38, 541-545.

Boria, R. A., Olson, L. E., Goodman, S. M. & Anderson, R. P. 2014. Spatial Filtering To Reduce Sampling Bias Can Improve The Performance Of Ecological Niche Models. Ecological Modelling, 275, 73-77.

Examples

```
data("sample_filt_data")

n.iterations <- 500

spatiotemp_thin(
 occ.data = sample_filt_data,
 temporal.method = "day",
 temporal.dist = 100,
 spatial.split.degrees = 3,
 spatial.dist = 100000,
 iterations = n.iterations
)
```

`spatiotemp_weights` *Calculate sampling effort across spatial and temporal buffer from species occurrence records*

Description

Calculates the total number of sampling events across a given spatial and temporal buffer from each occurrence record's co-ordinate and date.

Usage

```
spatiotemp_weights(
  occ.data,
  samp.events,
  spatial.dist = 0,
  temporal.dist = 0,
```

```
  prj = "+proj=longlat +datum=WGS84"
)
```

Arguments

occ.data	a data frame, with columns for occurrence record co-ordinates and dates with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
samp.events	a data.frame, sampling events with column names as follows; record longitude as "x", latitude as "y", year as "year", month as "month", and day as "day".
spatial.dist	a numeric value, the spatial distance in metres representing the radius from occurrence record co-ordinate to sum sampling events across.
temporal.dist	a numeric value, the temporal distance in days, representing the period before and after the occurrence record date to sum sampling events across.
prj	a character string, the coordinate reference system of input occ.data co-ordinates Default is "+proj=longlat +datum=WGS84".

Details

For each occurrence record, this function calculates the total number of sampling events within given radius (spatial.dist) from each record co-ordinate and days (temporal.dist) both prior and post record date.

In addition to total sampling events, the function also calculates relative sampling effort, scaling from 0 (least sampled) to 1 (most sampled).

Output could be used to calculate model weights to correct spatial and temporal biases in occurrence record collections (Stolar and Nielsen, 2015).

Value

Returns input occurrence record data frame with additional columns for sampling effort "SAMP_EFFORT" and relative sampling effort "REL_SAMP_EFFORT".

References

Stolar, J. & Nielsen, S. E. 2015. Accounting For Spatially Biased Sampling Effort In Presence-Only Species Distribution Modelling. Diversity And Distributions, 21, 595-608.

Examples

```
data("sample_explan_data")
data("sample_events_data")

spatiotemp_weights(
 occ.data = sample_explan_data,
 samp.events = sample_events_data,
 spatial.dist = 200000,
 temporal.dist = 20
```

)

Index