Package 'wallace'

March 9, 2024

Version 2.1.2

Date 2024-03-08

Title A Modular Platform for Reproducible Modeling of Species Niches and Distributions

Description The 'shiny' application Wallace is a modular platform for reproducible modeling of species niches and distributions. Wallace guides users through a complete analysis, from the acquisition of species occurrence and environmental data to visualizing model predictions on an interactive map, thus bundling complex workflows into a single, streamlined interface. An extensive vignette, which guides users through most package functionality can be found on the package's GitHub Pages website: <https://wallaceecomod.github.io/wallace/articles/tutorial-v2.html>.

Depends R (>= 3.5.0), shiny (>= 1.6.0), leaflet (>= 2.0.2)

Imports dplyr (>= 1.0.2), DT (>= 0.5), ecospat (>= 4.0.0), ENMeval (>= 2.0.3), knitcitations, leafem, leaflet.extras (>= 1.0.0), magrittr, markdown, methods, RColorBrewer, rJava, rlang, rmarkdown, sf, shinyalert, shinyjs, shinyWidgets (>= 0.6.0), spocc (>= 1.2.0), spThin, zip

Suggests ade4, BIEN, dismo, glue, jsonlite, knitr, mapview, maxnet, occCite, rangeModelMetadata, raster, rgbif (>= 3.3.0), sp, terra, testthat, tools

License GPL-3

URL http://wallaceecomod.github.io/wallace/,

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Jamie M. Kass [aut],
Gonzalo E. Pinilla-Buitrago [aut],
Andrea Paz [aut],
Bethany A. Johnson [aut],
Valentina Grisales-Betancur [aut],
Dean Attali [aut],

Matthew E. Aiello-Lammens [aut],
Cory Merow [aut],
Mary E. Blair [aut, cre],
Robert P. Anderson [aut],
Sarah I. Meenan [ctb],
Olivier Broennimann [ctb],
Peter J. Galante [ctb],
Brian S. Maitner [ctb],
Hannah L. Owens [ctb],
Sara Varela [ctb],
Bruno Vilela [ctb],
Robert Muscarella [ctb]

Maintainer Mary E. Blair <mblair1@amnh.org>

Repository CRAN

Date/Publication 2024-03-09 01:20:02 UTC

R topics documented:

`wallace-package` Wallace*: A modular platform for reproducible ecological modeling*

Description

Wallace is a `shiny` app that guides users through a complete species niche/distributional modeling analysis, from the acquisition of species occurrence and environmental data to visualizing model predictions on an interactive map (`rleaflet`), thus bundling complex workflows into a single, streamlined GUI interface. New functionality, in the form of modules, can be added to *Wallace* via contributions from the user community. In addition, executable session code (R Markdown format) can be downloaded to share with others or use as supplementary information for scientific papers and reports. The application is run via the function `run_wallace`.

Details

Please see the official website (`https://wallaceecomod.github.io/`) for more details. If you have questions about the application, please participate in the Google Group, or email the team directly: `<wallaceEcoMod@gmail.com>`.

`create_module` *Create a Wallace module*

Description

Create the template of a new Wallace module.

Usage

```
create_module(id, dir, map = FALSE, result = FALSE, rmd = FALSE, save = FALSE)
```

Arguments

`id`	The id of the module.
`dir`	A directory where the new module should be created.
`map`	Whether or not the module should support modifying the map.
`result`	Whether or not the module should support showing information in the Result tab.
`rmd`	Whether or not the module should add Rmd code to the Session Code download.
`save`	Whether or not the module has some custom data to save when the user saves the current session.

See Also

register_module

---

ecoClimate_getdata *ecoClimate_getdata*

Description

download ecoClimate layers. more info at www.ecoclimate.org

Usage

```
ecoClimate_getdata(AOGCM, Baseline, Scenario, logger)
```

Arguments

AOGCM	Select the AOGCM. Options are: "CCSM", "CNRM", "MIROC", "COSMOS", "FGOALS", "GISS", "IPSL", "MRI", "MPI"
Baseline	Select a baseline for the climatic layers. Options are: "Pre-industrial" (piControl-1760), "Historical" (1900-1949), "Modern" (1950-1999)
Scenario	Select a temporal scenario. Options are: "LGM" (21,000 years ago), "Holo" (6,000 years ago), "Present", "Future 2.6" (rcp 2.6), "Future 4.5" (rcp 4.5), "Future 6" (rcp 6), "Future 8.5" (rcp 8.5)
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL

Examples

```
## Not run:
CCSM_mod_present <- ecoclimate_getdata("CCSM", "Modern", "Present")
dev.new()
plot(CCSM_mod_present)

## End(Not run)
```

ecoClimate_select *ecoClimate_select*

Description

select which bioclimatic variables and set the extent you want (crop the raster stack to your study extent)

Usage

```
ecoClimate_select(map_climate, Sels=c(1:19), extent=c(-180, 180, -90, 90))
```

Arguments

map_climate	raster stack with all the variables
Sels	vector of integer numbers. 1 for bio1, 2 for bio2, etc. e.g. Sels= c(1,12,6) for selecting bio1, bio12 and bio6
extent	vector. xmin, xmax, ymin, ymax. e.g. c()

Examples

```
## Not run:
CCSM_mod_present <- ecoclimate_getdata("CCSM", "Modern", "Present")
Europe_CCSM_m_p_bio1_12 <- ecoClimate_select(CCSM_mod_present, c(1, 12),
                                             extent = c(-20, 80, 20, 80))
dev.new()
plot(Europe_CCSM_m_p_bio1_12)

## End(Not run)
```

envs_ecoClimate *envs_ecoClimate Obtain ecoClimate variables*

Description

download ecoClimate variables. See www.ecoclimate.org.

Usage

```
envs_ecoClimate(bcAOGCM, bcScenario, ecoClimSel, logger = NULL)
```

Arguments

bcAOGCM	Name of the Atmospheric and Oceanic Global Circulation Model. Options are: "CCSM", "CNRM", "MIROC", "FGOALS", "GISS", "IPSL","MRI", "MPI"
bcScenario	Select the temporal scenario that you want to download. Options are: "LGM" (21,000 years ago), "Holo" (6,000 years ago), "Present", "Future 2.6" (rcp 2.6), "Future 4.5" (rcp 4.5), "Future 6" (rcp 6), "Future 8.5" (rcp 8.5)
ecoClimSel	Numeric vector with list of variables to select.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.

Details

This function is called by the module envs to download ecoClimate variables from www.ecoclimate.org. The variables to be downloaded are selected by the user with bcSel and the resolution is fixed to 0.5 degrees. This function currently gets variables from Dropbox and the process takes significantly more time than for other datasets. It returns a rasterStack of selected variables.

Value

A rasterStack of selected variables

Author(s)

Sara Varela <sara_varela@yahoo.com>

Jamie M. Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
bcAOGCM <- "CCSM"
bcScenario <- "LGM"
ecoClimSel <- c(1,2,3)
## Not run:
varsEcoClimate <- envs_ecoClimate(bcAOGCM, bcScenario, ecoClimSel)

## End(Not run)
```

---

envs_userEnvs *envs_userEnvs*

Description

Load user provided rasters

Usage

```
envs_userEnvs(rasPath, rasName, doBrick = FALSE, logger = NULL)
```

Arguments

rasPath	character. Path to rasters, must be the full path including file name and extension
rasName	character. Vector of raster names to be assigned to loaded rasters
doBrick	logical. Converts downloaded rasters to brick for faster processing
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL

Details

This function is called by the module envs to load user provided raster variables for use in further analyses. It returns either a rasterStack or rasterBrick of loaded variables with appropriate names for further analyses.

Value

A rasterStack or a rasterBrick (if doBrick = TRUE) of user provided rasters

Author(s)

Jamie Kass <jamie.m.kass@gmail.com >

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
## Not run:
pathRast <- list.files(system.file("extdata/wc", package = "wallace"),
                       pattern = ".tif$", full.names = TRUE)
nameRast <- list.files(system.file("extdata/wc", package = "wallace"),
                       pattern = ".tif$", full.names = FALSE)
userEnvs <- envs_userEnvs(rasPath = pathRast, rasName = nameRast)

## End(Not run)
```

---

envs_worldclim *envs_worldclim Obtain WorldClim variables*

Description

download WorldClim variables. See www.worldclim.com.

Usage

```
envs_worldclim(bcRes, bcSel, mapCntr, doBrick = FALSE, logger = NULL)
```

Arguments

bcRes	numeric. Resolution of the climatic layers. Currently available resolutions are 0.5, 2.5 and 10.
bcSel	character. Vector with bionames to be selected.
mapCntr	numeric. Vector with longitude and latitude for a tile. Required for bcRes 0.5, for other resolutions world data will be downloaded.
doBrick	logical. Converts downloaded rasters to brick for faster processing.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.

Details

This function is called by the module envs to download WorldClim variables from www.worldclim.com. The variables to be downloaded are selected by the user with bcSel and the resolution with bcRes. It returns either a rasterStack or rasterBrick of selected variables with appropriate names for further analyses.

Value

A rasterStack or a rasterBrick (if doBrick=TRUE) of downloaded worldclim rasters at the requested resolution.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

getData

Examples

```
## Not run:
bcRes <- 10 # (10 arcmin)
envar <- c('bio05', 'bio06', 'bio13', 'bio14')
arcmin10 <- envs_worldclim(bcRes, bcSel = envar)

## End(Not run)
```

`espace_nicheOv` *espace_nicheOv Niche Overlap*

Description

Function evaluates niche overlap between the two species for which the occurrence density grid was computed

Usage

```
espace_nicheOv(
  z1,
  z2,
  iter = 100,
  equivalency = FALSE,
  similarity = TRUE,
  logger = NULL
)
```

Arguments

`z1`	ecospat niche object for species 1 from espace_occDens.
`z2`	ecospat niche object for species 2 from espace_occDens.
`iter`	numeric. Number of iterations.
`equivalency`	logical. Whether to run equivalency test. Default is FALSE.
`similarity`	logical. Whether to run similarity test. Default is TRUE.
`logger`	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.

Details

The niche overlap quantification is based on the occurrence densities and the densities of environmental conditions available in the background extent that are estimated in the module Occurrence Density Grid. The function computes 4 different things; Schoener's D, unfilling, stability, expansion indices (Guisan et al. 2014 TREE), and tests for niche equivalency and niche similarity.

Value

A list of 4 elements if all is set to TRUE. Elements are overlap (Schoener's D), USE (ecospat.niche.dyn.index), equiv and simil.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Olivier Broennimann <olivier.broennimann@unil.ch>

See Also

espace_pca espace_occDens ecospat.niche.overlap ecospat.niche.dyn.index ecospat.niche.equivalency.test ecospat.niche.similarity.test

Examples

```
## Not run:
sp.name1 <- "Bassaricyon_alleni"
sp.name2 <- "Bassaricyon_neblina"
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))

occs.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                    package = "wallace"))
occs.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                    package = "wallace"))

bgPts.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
                     package = "wallace"))
bgPts.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina_bgPoints.csv",
                     package = "wallace"))

occsExt.z1 <- raster::extract(envs, occs.z1[, c("longitude", "latitude")])
occsExt.z2 <- raster::extract(envs, occs.z2[, c("longitude", "latitude")])
bgExt.z1 <- raster::extract(envs, bgPts.z1[, c("longitude", "latitude")])
bgExt.z2 <- raster::extract(envs, bgPts.z2[, c("longitude", "latitude")])
pcaZ <- espace_pca(sp.name1, sp.name2,
                   occsExt.z1, occsExt.z2,
                   bgExt.z1, bgExt.z2)
occDens <- espace_occDens(sp.name1, sp.name2, pcaZ)
nicheOv <- espace_nicheOv(z1 = occDens[[sp.name1]],
                          z2 = occDens[[sp.name2]],
                          iter = 100, equivalency = TRUE,
                          similarity = TRUE)

## End(Not run)
```

---

espace_occDens *Occurrence density grid*

Description

calculates the part of environmental space more densely populated by species & the availability of environmental conditions in the background

Usage

```
espace_occDens(sp.name1, sp.name2, pca, logger = NULL)
```

Arguments

sp.name1	character name of species 1 to be analyzed.
sp.name2	character name of species 2 to be analyzed.
pca	pca output of pca component (in list format)
logger	stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.

Details

This function implements a density estimation for each region in the environmental space (gridded at 100*100 pixels). Then an occurrence density is estimated using a kernel density approach. The density of environmental conditions in the background is calculated in the same way.

Value

Returns a list of 2 lists (one for each species). Each list is an ecospat niche object that contains 10 species specific slots with information outputed by ecospat::grid.clim.dyn. z.uncor is the density of occurrence of the species and z.cor the occupancy of the environment by the species. It has the input parameters as individual slots.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com >

Olivier Broennimann <olivier.broennimann@unil.ch>

See Also

espace_pca espace_nicheOv ecospat.grid.clim.dyn

Examples

```
## Not run:
sp.name1 <- "Bassaricyon_alleni"
sp.name2 <- "Bassaricyon_neblina"
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))

occs.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                    package = "wallace"))
occs.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
```

```
                    package = "wallace"))

bgPts.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
                     package = "wallace"))
bgPts.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina_bgPoints.csv",
                     package = "wallace"))

occsExt.z1 <- raster::extract(envs, occs.z1[, c("longitude", "latitude")])
occsExt.z2 <- raster::extract(envs, occs.z2[, c("longitude", "latitude")])
bgExt.z1 <- raster::extract(envs, bgPts.z1[, c("longitude", "latitude")])
bgExt.z2 <- raster::extract(envs, bgPts.z2[, c("longitude", "latitude")])
pcaZ <- espace_pca(sp.name1, sp.name2,
                   occsExt.z1, occsExt.z2,
                   bgExt.z1, bgExt.z2)
occDens <- espace_occDens(sp.name1, sp.name2, pcaZ)

## End(Not run)
```

`espace_pca` *espace_pca Principal component analysis*

Description

Principal component analysis to reduce dimensionality of environmental space

Usage

```
espace_pca(
  sp.name1,
  sp.name2 = NULL,
  occs.z1,
  occs.z2 = NULL,
  bgPts.z1,
  bgPts.z2 = NULL,
  logger = NULL
)
```

Arguments

`sp.name1`	character. Name of species 1 to be analyzed.
`sp.name2`	character. Name of species 2 to be analyzed. Default is NULL.
`occs.z1`	table of occurrences with environmental values only for sp1.
`occs.z2`	table of occurrences with environmental values only for sp2. Default is NULL.
`bgPts.z1`	table of background points with environmental values only for sp1.
`bgPts.z2`	table of background points with environmental values only for sp2. Default is NULL.

`logger` Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL

Details

This function is called by the component espace to calibrate a PCA for 2 species in environmental space. When using within Wallace, GUI parameters are obtained from the model object, in particular, table of occurrences with environmental values and table of background points with environmental values. User must be careful as these tables must contain only environmental variables and not the point coordinates as outputted by model objects. The PCA is calibrated over the whole set of background points. The provided species name(s) are only used for logger messages and not for querying or selecting occurrences.

Value

A list of 14 elements of classes dudi and pca as in dudi.pca

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Olivier Broennimann <olivier.broennimann@unil.ch>

See Also

dudi.pca

Examples

```
## Not run:
sp.name1 <- "Bassaricyon_alleni"
sp.name2 <- "Bassaricyon_neblina"
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))

occs.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                    package = "wallace"))
occs.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                    package = "wallace"))

bgPts.z1 <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
                     package = "wallace"))
bgPts.z2 <- read.csv(system.file("extdata/Bassaricyon_neblina_bgPoints.csv",
                     package = "wallace"))

occsExt.z1 <- raster::extract(envs, occs.z1[, c("longitude", "latitude")])
occsExt.z2 <- raster::extract(envs, occs.z2[, c("longitude", "latitude")])
```

```
bgExt.z1 <- raster::extract(envs, bgPts.z1[, c("longitude", "latitude")])
bgExt.z2 <- raster::extract(envs, bgPts.z2[, c("longitude", "latitude")])
pcaZ <- espace_pca(sp.name1, sp.name2,
                   occsExt.z1, occsExt.z2,
                   bgExt.z1, bgExt.z2)

## End(Not run)
```

`model_bioclim` *model_bioclim Generate Bioclim model*

Description

The function generates a BIOCLIM model using ENMeval 2.0

Usage

```
model_bioclim(occs, bg, user.grp, bgMsk, logger = NULL, spN = NULL)
```

Arguments

`occs`	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
`bg`	coordinates of background points to be used for modeling.
`user.grp`	a list of two vectors containing group assignments for occurrences (occs.grp) and background points (bg.grp).
`bgMsk`	a RasterStack or a RasterBrick of environmental layers cropped and masked to match the provided background extent.
`logger`	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
`spN`	character. Species name to be used for all logger messages.

Details

The function generates a model in ENMeval using a user provided partition of occurrences from previous components in the GUI.

Value

Function returns an ENMevaluate object with all the evaluated models and a selection of appropriate fields.

Author(s)

Jamie M. Kass <jkass@gradcenter.cuny.edu>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

ENMevaluate

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))
bg <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
               package = "wallace"))
partblock <- part_partitionOccs(occs, bg, method = 'block')
m <- model_bioclim(occs, bg, partblock, envs)

## End(Not run)
```

---

model_maxent *model_maxent Generate maxent.jar or maxnet model*

Description

This functions generates maxent.jar or maxnet models using ENMeval 2.0 and user provided tuning parameters.

Usage

```
model_maxent(
  occs,
  bg,
  user.grp,
  bgMsk,
  rms,
  rmsStep,
  fcs,
  clampSel,
  algMaxent,
  catEnvs = NULL,
  parallel = FALSE,
  numCores = NULL,
  logger = NULL,
  spN = NULL
)
```

Arguments

occs	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
bg	coordinates of background points to be used for modeling.
user.grp	a list of two vectors containing group assignments for occurrences (occs.grp) and background points (bg.grp).
bgMsk	a RasterStack or a RasterBrick of environmental layers cropped and masked to match the provided background extent.
rms	vector of range of regularization multipliers to be used in the ENMeval run.
rmsStep	step to be used when defining regularization multipliers to be used from the provided range.
fcs	feature classes to be tested in the ENMeval run.
clampSel	Boolean use of clamping in the model.
algMaxent	character. algorithm to be used in modeling. A selection of "maxnet" or "maxent.jar".
catEnvs	if categorical predictor variables are included must provide the names.
parallel	logical. Whether to use parallel in the generation of models. Default is FALSE
numCores	numeric. If using parallel how many cores to use. Default is NULL.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	character. Species name to be used for all logger messages.

Details

The function generates model in ENMeval using a user provided partition of occurrences from previous components in the GUI. User can activate clamping and input tuning arguments to be used for model building.

Value

Function returns an ENMevaluate object with all the evaluated models and a selection of appropriate fields.

Author(s)

Jamie M. Kass <jkass@gradcenter.cuny.edu>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

ENMevaluate

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))
bg <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
               package = "wallace"))
partblock <- part_partitionOccs(occs, bg, method = 'block')
rms <- c(1:2)
rmsStep <- 1
fcs <- c('L', 'LQ')
m <- model_maxent(occs = occs, bg = bg, user.grp = partblock,
                  bgMsk = envs, rms = rms, rmsStep, fcs,
                  clampSel = TRUE, algMaxent = "maxnet",
                  parallel = FALSE)

## End(Not run)
```

---

occs_queryDb *occs_queryDb Query online database for species occurrence records.*

Description

Queries a given database for occurrence data on the provided species

Usage

```
occs_queryDb(
  spNames,
  occDb,
  occNum = NULL,
  doCitations = FALSE,
  gbifUser = NULL,
  gbifEmail = NULL,
  gbifPW = NULL,
  RmUncertain = FALSE,
  logger = NULL
)
```

Arguments

spNames	character. Species Latin name, with format "Genus species".

occDb	character. Biodiversity database to query; current choices are "gbif", "vertnet", and "BIEN"
occNum	numeric. Maximum number of occurrence records to return
doCitations	logical. Set TRUE to use ‘occCite‘ to get a complete list of original data sources in a citable format
gbifUser	specify only if using ‘occCite‘ with GBIF to get a complete list of original data sources in a citable format. This, as well as ‘gbifEmail‘ and ‘gbifPW‘ are constraints imposed by GBIF to obtain the complete set of metadata associated with occurrence records and is not stored or used by ‘wallace‘ for any other purposes.
gbifEmail	specify only if using ‘occCite‘ with GBIF to get a complete list of original data sources in a citable format.
gbifPW	specify only if using ‘occCite‘ with GBIF to get a complete list of original data sources in a citable format.
RmUncertain	specify if occurrences without uncertainty information should be removed (default is FALSE)
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL

Details

This function is called by the module occs_queryDb to query a database for species occurrence records, subset to only those records with coordinates, remove records with duplicate coordinates, and select some columns with fields appropriate to studies in biogeography.

Value

list of lists one list per species with occurrence records. Each individual species list with appropriate fields for analysis

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Hannah Owens

Andrea Paz <paz.andreita@gmail.com>

Examples

```
## Not run:
occs_queryDb(spName = "Bassaricyon alleni", occDb = "gbif", occNum = 10)

## End(Not run)
```

occs_userOccs *occs_userOccs Loads user provided occurrence records*

Description

Load user database with species occurrence records. Returns a list of lists, one per species provided in database in each species list with a set of appropriate fields

Usage

```
occs_userOccs(txtPath, txtName, txtSep = ",", txtDec = ".", logger = NULL)
```

Arguments

txtPath	path to database including database name and extension
txtName	name of database without the extension. Database must have at least three columns named 'scientific_name', 'longitude', 'latitude'
txtSep	field separator used in database (as in read.delim)
txtDec	decimal separator used for coordinates in database
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL

Details

This function is called by the module occs_queryDb to load a user provided database for species occurrence records, subset to only those records with coordinates, remove records with duplicate coordinates, and select some columns with fields appropriate to studies in biogeography.

Value

List of lists. One list per species with occurrence records. Each individual species list with appropriate fields for analysis

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
txtPath <- system.file("extdata/Bassaricyon_alleni.csv", package = "wallace")
txtName <- 'Bassaricyon_alleni'
user.occs <- occs_userOccs(txtPath, txtName)
```

part_partitionOccs *part_partitionOccs Partition occurrence data*

Description

This function partitions occurrence data and background points according to a user-selected method.

Usage

```
part_partitionOccs(
  occs,
  bg,
  method,
  kfolds = NULL,
  bgMask = NULL,
  aggFact = NULL,
  logger = NULL,
  spN = NULL
)
```

Arguments

occs	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
bg	coordinates of background points to be used for modeling.
method	character. Partitioning method to be used, one of 5 options: (1) 'jack' Non-spatial Partition - jackknife (2) 'rand' Non-spatial Partition - random k-fold (3) 'block' spatial Partition - block (4) 'cb1' spatial Partition - checkerboard 1 (K=2) (5) 'cb2' spatial Partition - checkerboard 2 (K=4)
kfolds	numeric. Number of partitions to create if selected method is random k-fold (must be >=2). If other method then keep default of NULL.
bgMask	a RasterStack or a RasterBrick of environmental layers cropped and masked.
aggFact	numeric. Aggregation factor to be used when using checkerboard partition (must be >= 1).
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is used in the partition occurrence data component. A user-selected method is used to partition occurrence and background points into different groups for model testing. A list of group assignments for both occurrences and background points is returned.

Value

A list of two vectors containing group assignments for occurrences (occs.grp) and background points (bg.grp).

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Andrea Paz <paz.andreita@gmail.com>

See Also

partitions

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))
bg <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
               package = "wallace"))
partblock <- part_partitionOccs(occs, bg, method = 'rand', kfold = 4)

## End(Not run)
```

---

penvs_bgExtent *penvs_bgExtent Generate background extent*

Description

This function generates a background area according to a user- provided method.

Usage

```
penvs_bgExtent(occs, bgSel, bgBuf, logger = NULL, spN = NULL)
```

Arguments

occs	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
bgSel	character. Method of background building. Must be one of three options: 'bounding box' , 'point buffers' or ' minimum convex polygon'.
bgBuf	numeric. Buffer distance in degrees to be used in the building of the background area.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is used in the select study region component. Here, the user can select between three methods ('bounding box', 'point buffers' or ' minimum convex polygon') to determine the background extent based on the observed occurrences. The function returns a SpatialPolygonsDataFrame object of the desired extent.

Value

A SpatialPolygons object that contains all occurrences from occs

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

penvs_userBgExtent, penvs_drawBgExtent, penvs_bgMask , penvs_bgSample

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
bgExt <- penvs_bgExtent(occs, bgSel = 'bounding box', bgBuf = 0.5)
```

penvs_bgMask *penvs_bgMask Mask environmental data*

Description

This functions crops and masks the environmental data to the provided background area.

Usage

```
penvs_bgMask(occs, envs, bgExt, logger = NULL, spN = NULL)
```

Arguments

occs	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
envs	a RasterStack or RasterBrick of environmental layers to be processed. This determines the output type.
bgExt	a SpatialPolygonsDataFrame with the background area to be used for processing.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	species name to be used for all logger messages

Details

This function is used in the select study region component. Here, the environmental layers to be used in the modeling are cropped and masked to the provided background area. The background area is determined in the function penvs_bgExtent from the same component. The function returns the provided environmental layers cropped and masked in the provided format (either a rasterBrick or a rasterStack).

Value

A RasterStack or a RasterBrick of environmental layers cropped and masked to match the provided background extent.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

penvs_userBgExtent, penvs_drawBgExtent, penvs_bgExtent, penvs_bgSample

Examples

```
## Not run:
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
bgExt <- penvs_bgExtent(occs, bgSel = 'bounding box', bgBuf = 0.5)
bgMask <- penvs_bgMask(occs, envs, bgExt)

## End(Not run)
```

`penvs_bgSample` *penvs_bgSample Sample background points*

Description

This function samples background points from an area determined by a rasterBrick or RasterStack of environmental layers previously cropped and masked to user determined extent.

Usage

```
penvs_bgSample(occs, bgMask, bgPtsNum, logger = NULL, spN = NULL)
```

Arguments

`occs`	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
`bgMask`	a RasterStack or a RasterBrick of environmental layers cropped and masked.
`bgPtsNum`	numeric. Number of points to be sampled from the area, they will be sampled as long as <= non NA cells in any reference layer.
`logger`	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
`spN`	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is used in the select study region component. Here, a user provided amount of points is randomly sampled from the RasterBrick or RasterStack of environmental variables cropped and masked to a given background extent. The maximum number of points to be sampled is the number of non NA cells in each layer of the reference RasterBrick or RasterStack If the requested number of points is larger than the number of cells in the reference RasterBrick or RasterStack then only a proportion of the requested will be returned.

Value

a dataframe containing point coordinates (longitude and latitude). All points are within the area provided in the RasterBrick or RasterStack (bgMask). Maximum number of points is equal to non NA cells in each layer of the reference brick or stack.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

penvs_bgMask, penvs_bgExtent penvs_userBgExtent, penvs_drawBgExtent, randomPoints

Examples

```
## Not run:
occs <-  occs_queryDb(spName = "Panthera onca", occDb = "gbif",
                      occNum = 100)
occs <- as.data.frame(occs[[1]]$cleaned)
envs <- envs_worldclim(bcRes = 10,
                       bcSel = c("bio03", "bio04", "bio13", "bio14"),
                       doBrick = TRUE)
bgExt <- penvs_bgExtent(occs, bgSel = 'bounding box', bgBuf = 0.5)
bgMask <- penvs_bgMask(occs, envs, bgExt)
bgsample <- penvs_bgSample(occs, bgMask, bgPtsNum = 1000)

## End(Not run)
```

---

penvs_drawBgExtent *penvs_drawBgExtent: Draw background extent*

Description

This function generates a background area according to a user drawn polygon and provided buffer.

Usage

```
penvs_drawBgExtent(
  polyExtXY,
  polyExtID,
  drawBgBuf,
  occs,
  logger = NULL,
  spN = NULL
)
```

Arguments

polyExtXY	coordinates of polygon endpoints obtained from user drawn polygon in GUI.
polyExtID	numeric. ID to be used in the generation of the polygon.
drawBgBuf	the buffer to be used in generating the SpatialPolygonsDataFrame, maybe be 0 or >0. A number must be specified.
occs	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is used in the select study region component. Here, in the GUI, the user draws a polygon to be used as the background extent and may include a buffer to the given polygon. The buffered poylgon must include all occurrences (occs) or function will return an error. The function returns a SpatialPolygonsDataFrame object of the desired extent (+ buffer).

Value

This functions returns a SpatialPolygons object based on the user specified coordinates (drawn on map). This SpatialPolygons object may be larger than specified if drawBgBuf > 0. The SpatialPolygons object will include all occurrences.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

penvs_userBgExtent, penvs_bgExtent, penvs_bgMask , penvs_bgSample

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
longitude <- c(-27.78641, -74.09170, -84.01930, -129.74867,
               -142.19085, -45.55045, -28.56050)
latitude <- c(-40.40539, -37.02010, 2.28455, 40.75350,
              56.35954, 54.55045, -7.11861)
expertDrawPoly <- matrix(c(longitude, latitude), byrow = FALSE,
                         ncol = 2)
drawBgBf <- penvs_drawBgExtent(polyExtXY = expertDrawPoly, polyExtID = 1,
                               drawBgBuf = 0.5, occs)
```

`penvs_userBgExtent` *penvs_userBgExtent: user provided background extent*

Description

This function generates a background area according to a user provided polygon and buffer.

Usage

```
penvs_userBgExtent(
  bgShp_path,
  bgShp_name,
  userBgBuf,
  occs,
  logger = NULL,
  spN = NULL
)
```

Arguments

`bgShp_path`	path to the user provided shapefile or csv with vertex coordinates.
`bgShp_name`	name of the user provided shapefile or csv with vertex coordinates.
`userBgBuf`	buffer to be used in creating the background extent must be >= 0.
`occs`	data frame of cleaned or processed occurrences obtained from components occs: Obtain occurrence data or, poccs: Process occurrence data.
`logger`	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
`spN`	Species name.

Details

This function is used in the select study region component. Here, the user provides either a shapefile or a csv with vertex coordinates with the desired shape for the background extent, the user may include a buffer to the given polygon. The buffered polygon must include all occurrences (occs) or function will return an error. The function returns a SpatialPolygons object of the desired extent (+ buffer).

Value

This function returns a SpatialPolygons object with the user provided shape (+ a buffer if userBgBuf >0). The polygon will be at least large enough to contain all occurrences.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Andrea Paz <paz.andreita@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

penvs_drawBgExtent, penvs_bgExtent, penvs_bgMask , penvs_bgSample

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                 package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
pathShp <- list.files(system.file("extdata/shp", package = "wallace"),
                      full.names = TRUE)
nameShp <- list.files(system.file("extdata/shp", package = "wallace"),
                      full.names = FALSE)
userBgbf <- penvs_userBgExtent(bgShp_path = pathShp, bgShp_name = nameShp,
                               userBgBuf = 0.2, occs = occs)
```

---

poccs_removeByID *poccs_removeByID Remove occurrence by ID*

Description

This function removes user selected occurrences by ID.

Usage

```
poccs_removeByID(occs, removeID, logger = NULL, spN = NULL)
```

Arguments

occs	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data
removeID	the ID of the occurrence to be removed from the occurrences dataframe.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is called by the remove occurrences by ID module. It allows for removal of a single occurrence flagged by the user on the map. The function will return a data frame of occurrences with all relevant columns for further analyses and without the occurrence selected by the user.

Value

A new occurence dataframe without the user selected occurrence mantaining all columns from original dataframe for further analyses.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                 package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
out.ID <- poccs_removeByID(occs, 11)
```

---

poccs_selectOccs *poccs_selectOccs Remove occurrences outside of polygon*

Description

This function removes occurrences outside of a user created polygon.

Usage

```
poccs_selectOccs(occs, polySelXY, polySelID = 1, logger = NULL, spN = NULL)
```

Arguments

occs	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data.
polySelXY	matrix of longitude and latitude describing the expert drawn polygon.
polySelID	numeric. Polygon ID to be used in SpatialPolygons creation, defaults to 1.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is called by the select occurrences on map module. It allows for removal of occurrences outside the user drawn polygon in the map. The function will return a data frame of occurrences with all relevant columns for further analyses and without the occurrences outside of the polygon.

Value

A new occurence dataframe including only occurences inside the provided polygon and mantaining all columns from original dataframe for further analyses.

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                             package = "wallace"))[, 2:3]
occs$occID <- 1:nrow(occs)
longitude <- c(-71.58400, -78.81300, -79.34034, -69.83331,
               -66.47149, -66.71319, -71.11931)
latitude <- c(13.18379, 7.52315, 0.93105, -1.70167,
              0.98391, 6.09208, 12.74980)
expertAddedPoly <- matrix(c(longitude, latitude), byrow = FALSE, ncol = 2)
out.occs <- poccs_selectOccs(occs, polySelXY = expertAddedPoly,
                             polySelID = 1)
```

poccs_thinOccs *poocs_thinOccs Thin occurrences*

Description

The function thins the observed occurrences by a user provided distance.

Usage

```
poccs_thinOccs(occs, thinDist, logger = NULL, spN = NULL)
```

Arguments

occs	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data
thinDist	distance in kilometers to be used for thinning. Number must be positive.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is called by the component poccs: process occurrence data to thin the occurrence data to a user specified distance. Providing an output with preserved columns appropriate for further analyses and a maximized number of occurrences that are separated by at least the provided distance.

Value

Output is a data frame of thinned occurences (all occurences at a distance >thinDist) with the same columns as occs

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

thin

Examples

```
occs <- read.csv(system.file("extdata/Bassaricyon_neblina.csv",
                             package = "wallace"))
occs$occID <- 1:nrow(occs)
out.thin <- poccs_thinOccs(occs = occs, thinDist = 30)
```

`register_module` *Register a Wallace module*

Description

Before running the Wallace application with `run_wallace()`, you can register your own modules to be used in Wallace.

Usage

```
register_module(config_file)
```

Arguments

`config_file`	The path to a YAML file that contains the information about one or more modules.

See Also

`create_module`

`run_wallace` *Run* Wallace *Application*

Description

This function runs the *Wallace* application in the user's default web browser.

Usage

```
run_wallace(launch.browser = TRUE, port = getOption("shiny.port"))
```

Arguments

`launch.browser`	Whether or not to launch a new browser window.
`port`	The port for the shiny server to listen on. Defaults to a random available port.

Note

Please see the official website (https://wallaceecomod.github.io/) for more details. If you have questions about the application, please participate in the Google Group, or email the team directly: <wallaceEcoMod@gmail.com>.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Examples

```
if(interactive()) {
run_wallace()
}
```

`vis_bioclimPlot` *vis_bioclimPlot Visualize bivariate plot of BIOCLIM model*

Description

This functions creates a bivariate plot with two of the environmental variables used for modeling as x and y axes and occurrences as observations.

Usage

```
vis_bioclimPlot(x, a = 1, b = 2, p = 0.9)
```

Arguments

x	bioclim model including values for each environmental layer at each occurrence point
a	numeric Environmental layer to be used as x axis. Default is layer 1.
b	numeric. Environmental layer to be used as x axis. Default is layer 2.
p	numeric. (0-1) percentile distribution to be used for plotting envelope and showing points outside of envelope. Default is 0.9

Details

This is a bivariate plot with x and y axes representing two of the environmental layers used for modeling (user selected although 1 and 2 as default). Occurrences used for modeling are shown with differential visualization if they are outside of the selected percentile distribution (for any variable). Plot also includes a rectangle representing the bivariate bioclimatic envelope according to a provided percentile.

Value

A bivariate plot of environmental values for occurrences. Includes a blue rectangle representing the bioclimatic envelope given p. Occurrences that are inside the envelope for all layers (included those not plotted) are shown as green circles and those outside of the envelope for one ore more variables are plotted as orange triangles.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

model_bioclim ENMevaluate

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
occs <- read.csv(system.file("extdata/Bassaricyon_alleni.csv",
                 package = "wallace"))
bg <- read.csv(system.file("extdata/Bassaricyon_alleni_bgPoints.csv",
               package = "wallace"))
partblock <- part_partitionOccs(occs, bg, method = 'block')
m <- model_bioclim(occs, bg, partblock, envs)
bioclimPlot <- vis_bioclimPlot(x = m@models$bioclim,
                               a = 1, b = 2, p = 1)

## End(Not run)
```

---

xfer_area *xfer_area Transfer model to a new area*

Description

Function transfers the model generated in previous components to a new user drawn area.

Usage

```
xfer_area(
  evalOut,
  curModel,
  envs,
  xfExt,
  alg,
  outputType = NULL,
  clamp = NULL,
  logger = NULL,
  spN = NULL
)
```

Arguments

evalOut	ENMevaluate output from previous module and using any of the available algorithms.
curModel	If algorithm is maxent, model selected by user as best or optimal, in terms of feature class and regularization multiplier (e.g 'L_1'). Else must be 1.
envs	environmental layers to be used for transferring the model. They must match the layers used for generating the model in the model component.
xfExt	extent of the area to transfer the model. This is defined by the user in the map of the GUI and is provided as a SpatialPolygons object.
alg	character. modeling algorithm used in the model component. Can be one of : 'BIOCLIM', 'maxent.jar' or 'maxnet'.
outputType	output type to be used when algorithm is maxnet or maxent.jar.
clamp	logical. Whether transfer will be of clamped or unclamped model.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	Character used to obtain species name for logger messages

Details

This functions transfers the model created in previous components to a new area. The area of transfer is user provided in the map of the GUI. The model will be transferred to the new area as long as the environmental variables are available for the area. This function returns a list including the cropped environmental variables used for transferring and the transferred model.

Value

A list of two elements: xferExt and xferArea. The first is a RasterBrick or a RasterStack of the environmental variables cropped to the area of transfer. The second element is a raster of the transferred model with the specified output type.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Andrea Paz <paz.andreita@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

predict, xfer_time xfer_userEnvs

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                           package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
# extent of transfer
longitude <- c(-71.58400, -78.81300, -79.34034, -69.83331,
               -66.47149, -66.71319, -71.11931)
latitude <- c(13.18379, 7.52315, 0.93105,
              -1.70167, 0.98391, 6.09208, 12.74980)
selCoords <- matrix(c(longitude, latitude), byrow = FALSE, ncol = 2)
polyExt <-
  sp::SpatialPolygons(list(sp::Polygons(list(sp::Polygon(selCoords)),
                      ID = 1)))
# load model
m <- readRDS(system.file("extdata/model.RDS",
                         package = "wallace"))
modXfer <- xfer_area(evalOut = m, curModel = 1, envs,
                     outputType = 'cloglog', alg = 'maxent.jar',
                     clamp = TRUE, xfExt = polyExt)

## End(Not run)
```

---

xfer_draw *xfer_draw Draw extent of transfer*

Description

This function creates a polygon object from coordinates of user drawn poylgon in the GUI.

Usage

```
xfer_draw(polyXfXY, polyXfID, drawXfBuf, logger = NULL, spN = NULL)
```

Arguments

polyXfXY	coordinates of polygon endpoints obtained from user drawn polygon
polyXfID	numeric .ID to be used in the generation of the polygon
drawXfBuf	the buffer to be used in generating the SpatialPolygonsDataFrame, must be >=0 . A number must be specified.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL
spN	character. Used to obtain species name for logger messages

Details

This function is used in the transfer model component. In the GUI, the user draws a polygon to be used as the extent of transfer and may include a buffer to the given polygon. The function returns a SpatialPolygonsDataFrame object of the desired extent (+ buffer).

Value

This functions returns a SpatialPolygons object based on the user specified coordinates (drawn on map). This SpatialPolygonsDataFrame may be larger than specified if drawBgBuf > 0.

Author(s)

Gonzalo Pinilla <gepinillab@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

xfer_userEnvs

Examples

```
longitude <- c(-27.78641, -74.09170, -84.01930, -129.74867,
               -142.19085, -45.55045, -28.56050)
latitude <- c(-40.40539, -37.02010, 2.28455, 40.75350,
              56.35954, 54.55045, -7.11861)
userDrawPoly <- matrix(c(longitude, latitude), byrow = FALSE,
                       ncol = 2)
drawXfBuf <- 0.5
polyXfID <- 1
polygonTest <- xfer_draw(polyXfXY = userDrawPoly, polyXfID,
                         drawXfBuf)
```

xfer_mess *xfer_mess generate MESS map for transferred raster*

Description

This function generates a MESS map for the new variables for transferring based on variables and points used for modeling in previous components.

Usage

```
xfer_mess(occs, bg, bgMsk, xferExtRas, logger = NULL, spN = NULL)
```

Arguments

occs	a data frame of occurrences used for modeling and values of environmental variables for each point.
bg	a data frame of points used as background for modeling and values of environmental variables for each point.
bgMsk	a rasterBrick or rasterStack of environmental variables used for modeling. They must be cropped and masked to extent used in model training.
xferExtRas	a rasterStack or rasterBrick of environmental variables to be used for transferring.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	character. Used to obtain species name for logger messages

Details

This functions allows for the creation of a MESS map for the new provided variables for transferring. These variables are either user uploaded or selected from WorldClim database. MESS map is based on occurrence and background points used for generating the model and the environmental values at those points.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

mess, xfer_time xfer_userEnvs

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                            package = "wallace"),
                      pattern = ".tif$", full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                            package = "wallace"),
                      pattern = ".tif$", full.names = FALSE))
# load model
m <- readRDS(system.file("extdata/model.RDS",
                         package = "wallace"))
occsEnvs <- m@occs
bgEnvs <- m@bg
envsFut <- list.files(path = system.file('extdata/wc/future',
                                         package = "wallace"),
                      full.names = TRUE)
envsFut <- raster::stack(envsFut)
## run function
xferMess <- xfer_mess(occs = occsEnvs, bg = bgEnvs, bgMsk = envs,
                      xferExtRas = envsFut)

## End(Not run)
```

`xfer_time` *xfer_time Transfer model to a new time*

Description

Function transfers the model generated in previous components to a new time and area using provided layers.

Usage

```
xfer_time(
  evalOut,
  curModel,
  envs,
  xfExt,
  alg,
  outputType = NULL,
  clamp = NULL,
  logger = NULL,
  spN = NULL
)
```

Arguments

`evalOut`	ENMevaluate output from previous module and using any of the available algorithms.

curModel	if algorithm is maxent, model selected by user as best or optimal, in terms of feature class and regularization multiplier (e.g 'L_1'). Otherwise must be 1.
envs	environmental layers of different time to be used for transferring the model. They must match the layers used for generating the model in the model component.
xfExt	extent of the area to transfer the model. This is defined by the user in the map of the GUI and is provided as a SpatialPolygons object.
alg	modeling algorithm used in the model component. Can be one of: 'bioclim', 'maxent.jar' or 'maxnet'.
outputType	output type to be used when algorithm is maxnet or maxent.jar.
clamp	logical. Whether transfer will be of clamped or unclamped model.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	character. Used to obtain species name for logger messages.

Details

This functions allows transferring the model created in previous components to a new time and area. The area of transfer is user provided in the map of the GUI and the transfer time user selected. The model will be transferred to the new area and time as long as the environmental variables are available for the area. This function returns a list including the cropped environmental variables used for transferring and the transferred model.

Value

A list of two elements: xferExt and xferTime. The first is a RasterBrick or RasterStack of the environmental variables cropped to the area of transfer. The second element is a raster of the transferred model with the specified output type.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Andrea Paz <paz.andreita@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

predict, xfer_time xfer_userEnvs

Examples

```
## Not run:
envs <- envs_userEnvs(rasPath = list.files(system.file("extdata/wc",
                                                       package = "wallace"),
                                           pattern = ".tif$",
```

```
                                        full.names = TRUE),
                      rasName = list.files(system.file("extdata/wc",
                                                       package = "wallace"),
                                        pattern = ".tif$",
                                        full.names = FALSE))
## extent to transfer
# set coordinates
longitude <- c(-71.58400, -78.81300, -79.34034, -69.83331, -66.47149, -66.71319,
               -71.11931)
latitude <- c(13.18379, 7.52315, 0.93105, -1.70167, 0.98391, 6.09208, 12.74980)
# generate matrix
selCoords <- matrix(c(longitude, latitude), byrow = FALSE, ncol = 2)
polyExt <- sp::SpatialPolygons(list(sp::Polygons(list(sp::Polygon(selCoords)),
                                                 ID = 1)))
# load model
m <- readRDS(system.file("extdata/model.RDS",
                         package = "wallace"))
occsEnvs <- m@occs
bgEnvs <- m@bg
envsFut <- list.files(path = system.file('extdata/wc/future',
                                         package = "wallace"),
                      full.names = TRUE)
envsFut <- raster::stack(envsFut)
modXfer <- xfer_time(evalOut = m, curModel = 1,
                     envs = envsFut, alg = 'maxent.jar',
                     xfExt = polyExt, clamp = FALSE, outputType = 'cloglog')

## End(Not run)
```

---

xfer_userEnvs *xfer_userEnvs Transfer model to user specified area and time*

Description

The function transfers the model generated in previous components to user uploaded environmental variables.

Usage

```
xfer_userEnvs(
  evalOut,
  curModel,
  envs,
  xfExt,
  alg,
  outputType = NULL,
  clamp = NULL,
  logger = NULL,
  spN = NULL
)
```

Arguments

evalOut	ENMevaluate output from previous module and using any of the available algorithms.
curModel	if algorithm is maxent, model selected by user as best or optimal, in terms of feature class and regularization multiplier (e.g 'L_1'). Otherwise it must be 1.
envs	user provided environmental layers (in raster format) to be used for transferring.
xfExt	extent of the area to transfer the model. This must be provided by the user as a shapefile or as a SpatialPolygons object.
alg	modeling algorithm used in the model component. Can be one of: 'BIOCLIM', 'maxent.jar' or 'maxnet'.
outputType	output type to be used when algorithm is maxnet or maxent.jar.
clamp	logical. Whether transfer will be of clamped or unclamped model.
logger	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
spN	character. Used to obtain species name for logger messages.

Details

This functions allows transferring the model created in previous components to a new time and area provided by the user. The transferring time and area is user-provided. The model will be transferred to the new time and area as long as the environmental variables provided are available for the area and match the variables used for model building. This function returns a list including the cropped environmental variables used for transferring and the transferred model.

Author(s)

Jamie Kass <jkass@gradcenter.cuny.edu>

Andrea Paz <paz.andreita@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

See Also

predict, xfer_time xfer_userExtent

Examples

```
## Not run:
## extent to transfer
# set coordinates
longitude <- c(-71.58400, -78.81300, -79.34034, -69.83331, -66.47149, -66.71319,
               -71.11931)
latitude <- c(13.18379, 7.52315, 0.93105, -1.70167, 0.98391, 6.09208, 12.74980)
# generate matrix
selCoords <- matrix(c(longitude, latitude), byrow = FALSE, ncol = 2)
polyExt <- sp::SpatialPolygons(list(sp::Polygons(list(sp::Polygon(selCoords)),
```

```
                                                    ID = 1)))
# load model
m <- readRDS(system.file("extdata/model.RDS",
                         package = "wallace"))
envsFut <- list.files(path = system.file('extdata/wc/future',
                                         package = "wallace"),
                      full.names = TRUE)
envsFut <- raster::stack(envsFut)
### run function
modXfer <- xfer_userEnvs(evalOut = m, curModel = 1, envs = envsFut,
                         outputType = "cloglog", alg = "maxent.jar",
                         clamp = FALSE, xfExt = polyExt)

## End(Not run)
```

`xfer_userExtent` *xfer_userExtent: user provided extent of transfer*

Description

This function generates an area of transfer according to a user provided polygon and buffer.

Usage

```
xfer_userExtent(bgShp_path, bgShp_name, userBgBuf, logger = NULL, spN = NULL)
```

Arguments

`bgShp_path`	path to the user provided shapefile or csv with vertex coordinates.
`bgShp_name`	name of the user provided shapefile or csv with vertex coordinates.
`userBgBuf`	numeric. Buffer to be used in creating the background extent must be >= 0.
`logger`	Stores all notification messages to be displayed in the Log Window of Wallace GUI. Insert the logger reactive list here for running in shiny, otherwise leave the default NULL.
`spN`	data frame of cleaned occurrences obtained from component occs: Obtain occurrence data. Used to obtain species name for logger messages.

Details

This function is used in the transfer component. Here, the user provides either a shapefile or a csv with vertex coordinates with the desired shape for the extent of transfer, the user may include a buffer to the given polygon. The function returns a SpatialPolygons object of the desired extent (+ buffer).

Value

This function returns a SpatialPolygons object with the user provided shape (+ a buffer is userBgBuf >0).

Author(s)

Jamie Kass <jamie.m.kass@gmail.com>

Gonzalo E. Pinilla-Buitrago <gepinillab@gmail.com>

Andrea Paz <paz.andreita@gmail.com>

Bethany A. Johnson <bjohnso005@citymail.cuny.edu>

See Also

penvs_drawBgExtent, penvs_bgExtent, penvs_bgMask , penvs_bgSample

Examples

```
pathShp <- list.files(system.file("extdata/shp", package = "wallace"),
                      full.names = TRUE)
nameShp <- list.files(system.file("extdata/shp", package = "wallace"),
                      full.names = FALSE)
xferUser <- xfer_userExtent(bgShp_path = pathShp, bgShp_name = nameShp,
                            userBgBuf = 1)
```

Index