Package ‘crestr’

July 30, 2021

Title A Probabilistic Approach to Reconstruct Past Climates Using Biological Climate Proxies

Version 0.0.0.9000

Description This package was designed to apply the CREST climate reconstruction
method. It can be used using the calibration data that can be obtained
through the package or by importing private data. An ensemble of
graphical outputs were designed to facilitate the use of the
package and the interpretation of the results.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports DBI,
RPostgreSQL,
graphics,
plot3D,
viridis,
grDevices,
methods,
stringr,
raster,
plyr,
sp,
rgdal,
RPostgres,
rgeos,
scales,
pals

Depends R (>= 2.10)

Suggests testthat,
knitr,
rmarkdown,
clipr,
openxlsx

VignetteBuilder knitr

URL `https://github.com/mchevalier2/crestr`, `https://mchevalier2.github.io/crestr/index.html`

BugReports https://github.com/mchevalier2/crestr/issues

R topics documented:

`accBasinNames` *Return the list of oceans and seas.*

Description

Return the list of oceans and seas.

Usage

```
accBasinNames(basin = NA)
```

Arguments

`basin`	A name of basin. Default is `NA` and returns a list of all the accpted names.

Value

A list of accepted names.

Examples

```
accBasinNames()
accBasinNames('Indian Ocean')
```

`accClimateVariables` *Describes all the variables available in the database.*

Description

Provides the index and the short and full names of all the variables available in the database.

Usage

```
accClimateVariables(v = NA, domain = NA)
```

Arguments

`v`	The name of a variable to quickly access its description and ID (default `NA` returns all possible values).
`domain`	The domain ('Terrestrial' or 'Marine') of the variables. Default value is *NA* and both terrestrial and marine climate variable names are returned.

Value

A data frame descriptive of the climate variables available in the database (if `v=NA`) or the description of variable v.

Examples

```
accClimateVariables()
accClimateVariables(v='bio12')
```

`accCountryNames` *Return the list of the continents and associated countries.*

Description

Return the list of the continents and associated countries.

Usage

```
accCountryNames(continent = NA)
```

Arguments

`continent`	A name of continent. Default is `NA` and returns a list of all the country names sorted by continent.

Value

A list where each element is a vector of corresponding country names.

Examples

```
accCountryNames()
accCountryNames('Europe')
```

accRealmNames *Return the list of the realms and associated biomes and ecoregions.*

Description

Return the list of the realms and associated biomes and ecoregions.

Usage

```
accRealmNames(realm = NA, ecoregion = TRUE)
```

Arguments

`realm`	A name of accepted realm. Default is `NA` and returns a list of all the biome and ecoregion names sorted by realm.
`ecoregion`	A boolean to choose whether to get the ecoregions names.

Value

A list with elements that correspond to the biomes (and possibly ecoregions) of each realm.

Examples

```
accRealmNames()
accRealmNames(realm='Africotropical')
accRealmNames(realm='Africotropical', ecoregion=FALSE)
```

calib_clim_space *Calibrate the distribution of the modern climate space.*

Description

Calibrate the distribution of the modern climate space.

Usage

```
calib_clim_space(climate, bin_width)
```

Arguments

`climate`	A vector of climatic values where the species is present.
`bin_width`	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.

Value

A ccs object that will be used by `fit_pdfsp`.

Examples

```
calib_clim_space(sample(0:300 / 10, 4000, replace = TRUE), 2)
```

`check_coordinates` *Check if the coordinates are correct.*

Description

Check if the coordinates are correct.

Usage

```
check_coordinates(xmn, xmx, ymn, ymx)
```

Arguments

`xmn`	The coordinates defining the study area.
`xmx`	The coordinates defining the study area.
`ymn`	The coordinates defining the study area.
`ymx`	The coordinates defining the study area.

Examples

```
check_coordinates(NA, NA, NA, NA)
check_coordinates(-200, 0, 0, 90)
check_coordinates(20, 0, 90, 0)
```

`cite_climate_data` *Returns the references associated with the climate data used to fit the* `pdfs`.

Description

Returns the references associated with the climate data used to fit the `pdfs`.

Usage

```
cite_climate_data(x, verbose = TRUE)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

`cite_crest` *Returns the list of references associated to the reconstruction.*

Description

Returns the list of references associated to the reconstruction.

Usage

```
cite_crest(x, verbose = TRUE)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

`cite_distrib_data` *Returns the references associated with the GBIF data used to fit the* `pdfs`.

Description

Returns the references associated with the GBIF data used to fit the `pdfs`.

Usage

```
cite_distrib_data(x, verbose = TRUE)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

`cite_method` *Returns the references associated with the development of CREST.*

Description

Returns the references associated with the development of CREST.

Usage

```
cite_method(x, verbose = TRUE)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

`climate_from_xy` *Extract The climate values associated to a set of coordinates.*

Description

Extract The climate values associated to a set of coordinates.

Usage

```
climate_from_xy(
  long,
  lat,
  climate = accClimateVariables()[, 2],
  resol = 0.25,
  dbname = "gbif4crest_02"
)
```

Arguments

`long`	The longitude of the site.
`lat`	The latitude of the site.
`climate`	The climate variables to extract the values from. Returns all possible values by default.
`resol`	The resolution of the target climatology (default 0.25 degrees).
`dbname`	The name of the data source database.

Value

A data frame containing the climate values.

Examples

```
climate_from_xy(50, 10, c('bio1', 'ai'))
climate_from_xy(50, 10)
```

`close_db_connection` *Disconnect the database connection.*

Description

Disconnect the database connection.

Usage

```
close_db_connection(db)
```

Arguments

`db`	An active database connection

Examples

```
db <- connect_online()
close_db_connection(db)
## Not run:
  db <- connect_online()
  close_db_connection(db)

## End(Not run)
```

`colour_theme` *Returns a vector of colours*

Description

Returns a vector of colours

Usage

```
colour_theme(n)
```

Arguments

`n`	An idex to select the colour theme

Examples

```
colour_theme(1)
```

`connect_online` *Connect to the gbif4crest calibration database*

Description

Connect to the gbif4crest_02 database by accessing the server on Amazon.

Usage

```
connect_online(
  dbname = "gbif4crest_02",
  port = 5432,
  host = "gbif4crest.cvqgy2mnjwtg.eu-west-3.rds.amazonaws.com",
  user = "guestuser",
  password = "pwd12345"
)
```

Arguments

`dbname`	The name of the database. Default is `'gbif4crest_02'`.
`port`	The port to connect to the server. Default is 5432.
`host`	The host of the database server. Default is `'gbif4crest.cvqgy2mnjwtg.eu-west-3.rds.amazonaw`
`user`	The user name to use to connect. Default is `'guestuser'`.
`password`	The password associated with the user name. Default is `'pwd12345'`.

Value

An active connection to a database

Examples

```
## Not run:
  db <- connect_online()

## End(Not run)
```

`convert2percentages` *Convert abundance data into percentage data.*

Description

Convert abundance data into percentage data.

Usage

```
convert2percentages(df, col2convert = 2:ncol(df))
```

Arguments

`df`	The dataframe containing the data to convert.
`col2convert`	A vector of the columns to convert. Default is all the columns but the first, which contains an age, a depth or a sampleID.

Value

A vector of unique taxonIDs.

Examples

```
df <- data.frame(matrix(1:25, ncol = 5))
colnames(df) <- paste(rep("col", 5), 1:5, sep = "")
convert2percentages(df)
convert2percentages(df, col2convert = 3:5)
```

`convert2presenceAbsence` *Convert data into presence/absence data.*

Description

Convert data into presence/absence data.

Usage

```
convert2presenceAbsence(df, threshold = 2, col2convert = 2:ncol(df))
```

Arguments

`df`	The dataframe containing the data to convert.
`threshold`	The threshold that defines presence (presence if >= threshold)
`col2convert`	A vector of the columns to convert. Default is all the columns but the first, which contains an age, a depth or a sampleID.

Value

A vector of unique taxonIDs.

Examples

```
df <- data.frame(matrix(1:25, ncol = 5))
colnames(df) <- paste(rep("col", 5), 1:5, sep = "")
convert2presenceAbsence(df, threshold = 15)
convert2presenceAbsence(df, col2convert = 3:5)
```

`copy_crest` *Copy crest data to the clipboard.*

Description

Copy crest data to the clipboard for an easy extraction of the data from the R environment.

Usage

```
copy_crest(
  x,
  climate = x$parameters$climate,
  optima = TRUE,
  mean = FALSE,
  uncertainties = FALSE
)
```

Arguments

x	A crestObj produced by the crest.reconstruct or crest functions.
climate	A vector of the climate variables to extract. See accClimateVariables for the list of accepted values.
optima	A boolean value to indicate if the optima should be copied to the clipboard.
mean	A boolean value to indicate if the means should be copied to the clipboard.
uncertainties	A boolean value to indicate if the uncertainties should be copied to the clipboard.

Examples

```
## Not run:
 if(requireNamespace('clipr', quietly=TRUE)) {
   reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example",
    leave_one_out = TRUE
   )
   copy_crest(reconstr, uncertainties=TRUE)
   ## You can now paste the values in a spreadsheet.
 }

## End(Not run)
```

createPSE *Returns the citations associated to the GBIF data used to fit the pdfs.*

Description

Returns the citations associated to the GBIF data used to fit the pdfs.

Usage

```
createPSE(taxa, loc = "proxy_species_equivalency.xlsx")
```

Arguments

taxa	A list of taxa to include in the PSE file.
loc	An absolute or relative path that indicates where the spreadsheet should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name proxy_species_equivalency.xlsx.

Examples

```
## Not run:
   data(crest_ex)
   createPSE(taxa = colnames(crest_ex)[-1])

## End(Not run)
```

crest — *A wrapper for all the crest functions.*

Description

Runs all the different steps of a CREST reconstruction in one function.

Usage

```
crest(
  df,
  climate,
  pse = NA,
  taxaType = 0,
  distributions = NA,
  site_info = rep(NA, length(climate)),
  site_name = NA,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  continents = NA,
  countries = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  minGridCells = 20,
  selectedTaxa = NA,
  bin_width = rep(1, length(x$parameters$climate)),
  shape = rep("normal", length(x$parameters$climate)),
  npoints = 500,
  ai.sqrt = FALSE,
  geoWeighting = TRUE,
  climateSpaceWeighting = TRUE,
  presenceThreshold = 0,
  taxWeight = "normalisation",
  uncertainties = c(0.5, 0.95),
  leave_one_out = FALSE,
  verbose = TRUE,
  dbname = "gbif4crest_02"
)
```

Arguments

`df`	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`pse`	A pollen-Species equivalency table. See `createPSE` for details.

`taxaType`	A numerical index (between 1 and 6) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents. The example dataset uses taxaType=0 (pseudo-data). Default is 1.
`distributions`	A dataframe containing the presence records of the studied proxies and their associated climate values.
`site_info`	A vector containing the coordinates of the study site. Default `c(NA,NA)`.
`site_name`	The name of the dataset (default `NA`).
`xmn`	The coordinates defining the study area.
`xmx`	The coordinates defining the study area.
`ymn`	The coordinates defining the study area.
`ymx`	The coordinates defining the study area.
`continents`	A vector of the continent names defining the study area.
`countries`	A vector of the country names defining the study area.
`realms`	A vector of the studied botanical realms defining the study area.
`biomes`	A vector of the studied botanical biomes defining the study area.
`ecoregions`	A vector of the studied botanical ecoregions defining the study area.
`minGridCells`	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
`selectedTaxa`	A data frame assigns which taxa should be used for each variable (1 if the taxon should be used, 0 otherwise). The colnames should be the climate variables' names and the rownames the taxa names. Default is 1 for all taxa and all variables.
`bin_width`	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.
`shape`	The imposed shape of the species `pdfs`. We recommend using 'normal' for temperature variables and 'lognormal' for the variables that can only take positive values, such as precipitation or aridity. Default is 'normal' for all.
`npoints`	The number of points to be used to fit the `pdfs`. Default 200.
`ai.sqrt`	A boolean to indicate whether ai values should be square-root transformed (default `FALSE`).
`geoWeighting`	A boolean to indicate if the species should be weighting by the squareroot of their extension when estimating a genus/family level taxon-climate relationships.
`climateSpaceWeighting`	A boolean to indicate if the species `pdfs` should be corrected for the modern distribution of the climate space (default `TRUE`).
`presenceThreshold`	All values above that threshold will be used in the reconstruction (e.g. if set at 1, all percentages below 1 will be set to 0 and the associated presences discarded). Default is 0.
`taxWeight`	One value among the following: 'originalData', 'presence/absence', 'percentages' or 'normalisation' (default).
`uncertainties`	A (vector of) threshold value(s) indicating the error bars that should be calculated (default both 50 and 95% ranges).

`leave_one_out`	A boolean to indicate whether the leave one out (loo) reconstructions should be computed (default `FALSE`).
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).
`dbname`	The name of the database. Default is `'gbif4crest_02'`.

Value

A `crestObj` containing the reconstructions.

Examples

```
data(crest_ex)
data(crest_ex_pse)
data(crest_ex_selection)
reconstr <- crest(
  df = crest_ex, pse = crest_ex_pse, taxaType = 0,
  site_info = c(7.5, 7.5), site_name = 'crest_example',
  climate = c("bio1", "bio12"), bin_width = c(2, 50),
  shape = c("normal", "lognormal"),
  selectedTaxa = crest_ex_selection, dbname = "crest_example",
  leave_one_out = TRUE,
  verbose = FALSE
)
plot(reconstr)
plot_loo(reconstr)
```

`crest.calibrate` *Fit the species and proxy* `pdfs`

Description

This function fits the climate response of the selected taxa to the selected climate variables.

Usage

```
crest.calibrate(
  x,
  bin_width = x$parameters$bin_width,
  shape = x$parameters$shape,
  npoints = x$parameters$npoints,
  geoWeighting = x$parameters$geoWeighting,
  climateSpaceWeighting = x$parameters$climateSpaceWeighting,
  verbose = TRUE
)
```

Arguments

`x`	A `crestObj` produced by the `crest.get_modern_data` function.
`bin_width`	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.

`shape`	The imposed shape of the species `pdfs`. We recommend using 'normal' for temperature variables and 'lognormal' for the variables that can only take positive values, such as precipitation or aridity. Default is 'normal' for all.
`npoints`	The number of points to be used to fit the `pdfs`. Default 200.
`geoWeighting`	A boolean to indicate if the species should be weighting by the squareroot of their extension when estimating a genus/family level taxon-climate relationships.
`climateSpaceWeighting`	A boolean to indicate if the species `pdfs` should be corrected for the modern distribution of the climate space (default `TRUE`).
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

Value

A `crestObj` object containing the spatial distributions and the climate space.

Examples

```
data(crest_ex_pse)
data(crest_ex_selection)
data(crest_ex)
x <- crest.get_modern_data( df = crest_ex,
  pse = crest_ex_pse, taxaType = 0,
  climate = c("bio1", "bio12"),
  selectedTaxa = crest_ex_selection, dbname = "crest_example",
  verbose = FALSE
)
x <- crest.calibrate(x,
  geoWeighting = TRUE, climateSpaceWeighting = TRUE,
  bin_width = c(2, 50), shape = c("normal", "lognormal"),
  verbose = FALSE
)
```

`crest.get_modern_data` *Extract distributions from the database*

Description

This function will extract the distributions of all the species composing each taxon and return them as a list.

Usage

```
crest.get_modern_data(
  pse,
  taxaType,
  climate,
  df = NA,
  ai.sqrt = FALSE,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  continents = NA,
  countries = NA,
  basins = NA,
  sectors = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  minGridCells = 20,
  elev_min = NA,
  elev_max = NA,
  elev_range = NA,
  year_min = 1900,
  year_max = 2021,
  nodate = TRUE,
  type_of_obs = c(1, 2, 3, 8, 9),
  selectedTaxa = NA,
  site_info = c(NA, NA),
  site_name = NA,
  dbname = "gbif4crest_02",
  verbose = TRUE
)
```

Arguments

`pse`	A pollen-Species equivalency table. See `createPSE` for details.
`taxaType`	A numerical index (between 1 and 6) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents. The example dataset uses taxaType=0 (pseudo-data). Default is 1.
`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`df`	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
`ai.sqrt`	A boolean to indicate whether ai values should be square-root transformed (default `FALSE`).
`xmn`	The coordinates defining the study area.
`xmx`	The coordinates defining the study area.
`ymn`	The coordinates defining the study area.
`ymx`	The coordinates defining the study area.
`continents`	A vector of the continent names defining the study area.
`countries`	A vector of the country names defining the study area.
`basins`	A vector of the ocean names defining the study area.
`sectors`	A vector of the marine sector names defining the study area.
`realms`	A vector of the studied botanical realms defining the study area.
`biomes`	A vector of the studied botanical biomes defining the study area.
`ecoregions`	A vector of the studied botanical ecoregions defining the study area.

`minGridCells`	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
`elev_min`	Parameters to only selected grid cells with an elevation higher than elev_min or lower than elev_max (default is 'NA).
`elev_max`	Parameters to only selected grid cells with an elevation higher than elev_min or lower than elev_max (default is 'NA).
`elev_range`	Parameters discard the grid cell with a high elevational range (default is NA).
`year_min`	The oldest and youngest occurrences accepted (default is 1900-2021).
`year_max`	The oldest and youngest occurrences accepted (default is 1900-2021).
`nodate`	A boolean to accept occurrences without a date (can overlap with occurrences with a date; default TRUE).
`type_of_obs`	The type of observation to use in the study. 1: human observation, 2: observation, 3: preserved specimen, 4: living specimem, 5: fossil specimen, 6: material sample, 7: machine observation, 8: literature, 9: unknown (Default `c(1,2,3,8,9)`)
`selectedTaxa`	A data frame assigns which taxa should be used for each variable (1 if the taxon should be used, 0 otherwise). The colnames should be the climate variables' names and the rownames the taxa names. Default is 1 for all taxa and all variables.
`site_info`	A vector containing the coordinates of the study site. Default `c(NA,NA)`.
`site_name`	The name of the dataset (default NA).
`dbname`	The name of the database. Default is `'gbif4crest_02'`.
`verbose`	A boolean to print non-essential comments on the terminal (default TRUE).

Value

A `crestObj` object containing the spatial distributions.

Examples

```
data(crest_ex_pse)
data(crest_ex_selection)
data(crest_ex)
x <- crest.get_modern_data( df = crest_ex,
  pse = crest_ex_pse, taxaType = 0,
  climate = c("bio1", "bio12"),
  selectedTaxa = crest_ex_selection, dbname = "crest_example",
  verbose = FALSE
)
x
lapply(x$modelling$distributions, head)
```

`crest.reconstruct` *Reconstruct climate from fossil data*

Description

This function fits the climate response of the selected taxa to the selected climate variables.

Usage

```
crest.reconstruct(
  x,
  presenceThreshold = 0,
  taxWeight = "normalisation",
  uncertainties = c(0.5, 0.95),
  skip_for_loo = FALSE,
  verbose = TRUE
)
```

Arguments

`x`	A `crestObj` produced by the `crest.calibrate` function.
`presenceThreshold`	All values above that threshold will be used in the reconstruction (e.g. if set at 1, all percentages below 1 will be set to 0 and the associated presences discarded). Default is 0.
`taxWeight`	One value among the following: 'originalData', 'presence/absence', 'percentages' or 'normalisation' (default).
`uncertainties`	A (vector of) threshold value(s) indicating the error bars that should be calculated (default both 50 and 95% ranges).
`skip_for_loo`	A boolean that tells the `loo` function to skip parts and fasten the process. Not for users, always leve to `FALSE`.
`verbose`	A boolean to print non-essential comments on the terminal (default `TRUE`).

Value

A `crestObj` object containing the reconstructions and all the associated data.

Examples

```
data(crest_ex_pse)
data(crest_ex_selection)
data(crest_ex)
x <- crest.get_modern_data( df = crest_ex,
  pse = crest_ex_pse, taxaType = 0,
  climate = c("bio1", "bio12"),
  selectedTaxa = crest_ex_selection, dbname = "crest_example",
  verbose = FALSE
)
x <- crest.calibrate(x,
  geoWeighting = TRUE, climateSpaceWeighting = TRUE,
  bin_width = c(2, 50), shape = c("normal", "lognormal"),
  verbose = FALSE
)
x <- crest.reconstruct(x,
  verbose = FALSE)
plot(x)
```

`crest.set_modern_data` *Format a* `crestObj` *with private data.*

Description

Format a `crestObj` with private data.

Usage

```
crest.set_modern_data(
  distributions,
  climate,
  df = NA,
  climate_space = NA,
  weight = FALSE,
  minGridCells = 0,
  selectedTaxa = NA,
  site_info = c(NA, NA),
  site_name = NA,
  site_climate = rep(NA, length(climate)),
  verbose = TRUE
)
```

Arguments

`distributions`	A dataframe containing the presence records of the studied proxies and their associated climate values.
`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`df`	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
`climate_space`	A dataframe of climate values across the study area useful to correct for a disbalanced sampling data (see '`crest.calibrate` for more details). Default is `NA`.
`weight`	The records in the distributions can be weighted using the percentages by setting weight=TRUE. Include a column called 'weight' in the `distributions` table.
`minGridCells`	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
`selectedTaxa`	A data frame assigns which taxa should be used for each variable (1 if the taxon should be used, 0 otherwise). The colnames should be the climate variables' names and the rownames the taxa names. Default is 1 for all taxa and all variables.
`site_info`	A vector containing the coordinates of the study site. Default `c(NA,NA)`.

site_name	The name of the dataset (default NA).
site_climate	The climate values at the location of the dataset '(default NA).
verbose	A boolean to print non-essential comments on the terminal (default TRUE).

Value

A crestObj object containing the spatial distributions.

Examples

```
#> Reformating the example dataset to fit this function
distributions <- cbind('ProxyName'= rep('Taxon1', nrow(reconstr$modelling$distributions[[1]])),
                       reconstr$modelling$distributions[[1]],
                       stringsAsFactors = FALSE)
for(tax in names(reconstr$modelling$distributions)[-1]) {
  distributions <- rbind(distributions,
                 cbind('ProxyName'= rep(tax, nrow(reconstr$modelling$distributions[[tax]])),
                           reconstr$modelling$distributions[[tax]],
                           stringsAsFactors = FALSE)
                     )
}
distributions <- distributions[, c(2,1,3:6)]
print(head(distributions))

climate_space <- reconstr$modelling$climate_space
print(head(climate_space))

x <- crest.set_modern_data(distributions, df=crest_ex,
                           climate = c("bio1", "bio12"))
x <- crest.set_modern_data(distributions, df=crest_ex,
                           climate_space=climate_space,
                           climate = c("bio1", "bio12"))
```

crest_ex *Example dataset to run the CREST method for the first time.*

Description

A dataset containing 20 randomly generated pollen samples for 7 pollen taxa.

Usage

```
crest_ex
```

Format

A data frame with 20 rows (samples) and 8 columns (1 column for the age and one for each of the 7 taxa):

Age: Age of each sample

Taxon1: Percentage of Taxon1 in each sample.

Taxon2: Percentage of Taxon2 in each sample.

Taxon3: Percentage of Taxon3 in each sample.

Taxon4: Percentage of Taxon4 in each sample.

Taxon5: Percentage of Taxon5 in each sample.

Taxon6: Percentage of Taxon6 in each sample.

Taxon7: Percentage of Taxon7 in each sample.

`crest_ex_pse` *Example dataset to Extract data from the example database.*

Description

A database indicating the taxonomy of the example proxies.

Usage

```
crest_ex_pse
```

Format

A data frame with 7 rows (taxa) and 5 columns (taxonomy description):

Level: An integr indicating the taxonomic resolution (1 family, 2 genus, 3 species, 4 or higher ignore taxon)

Family: The family corresponding to the ProxyName

Genus: The genus corresponding to the ProxyName

Species: The species corresponding to the ProxyName

ProxyName: The names of the observed proxies, as reported in the main data file

---

`crest_ex_selection` *Example dataset to associate taxa with climate varibles.*

Description

A data frame indicating the taxa that should be used to reconstruct each climate variable (1s in the matrix) and those who should be excluded (0s).

Usage

```
crest_ex_selection
```

Format

A data frame with 7 rows (taxa) and 2 columns (climate variables):

bio1: The first variable to reconstruct (mean annual temperature)

bio12: The second variable to reconstruct (annual precipitation)

`crestObj` *Create a* `crestObj` *object.*

Description

Creates a `crestObj` object with all default parameters.

Usage

```
crestObj(
  taxa.name,
  taxaType,
  climate,
  pse = NA,
  dbname = NA,
  continents = NA,
  countries = NA,
  basins = NA,
  sectors = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  elev_min = NA,
  elev_max = NA,
  elev_range = NA,
  year_min = 1900,
  year_max = 2021,
  nodate = TRUE,
  type_of_obs = c(1, 2, 3, 8, 9),
  df = NA,
  x = NA,
  x.name = "",
  minGridCells = 20,
  weightedPresences = FALSE,
  bin_width = NA,
  shape = NA,
  npoints = 200,
  geoWeighting = TRUE,
  climateSpaceWeighting = TRUE,
  selectedTaxa = NA,
  distributions = NA,
  presenceThreshold = 0,
  taxWeight = "normalisation",
  uncertainties = c(0.5, 0.95)
)
```

Arguments

`taxa.name`	A vector that contains the names of the taxa to study.
`taxaType`	A numerical index (between 1 and 6) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents. The example dataset uses taxaType=0 (pseudo-data). Default is 1.
`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`pse`	A pollen-Species equivalency table. See `createPSE` for details.
`dbname`	The name of the data source database.
`continents`	A vector of the continent names defining the study area.
`countries`	A vector of the country names defining the study area.
`basins`	A vector of the ocean names defining the study area.
`sectors`	A vector of the marine sector names defining the study area.
`realms`	A vector of the studied botanical realms defining the study area.
`biomes`	A vector of the studied botanical biomes defining the study area.
`ecoregions`	A vector of the studied botanical ecoregions defining the study area.
`xmn, xmx, ymn, ymx`	The coordinates defining the study area.
`elev_min, elev_max`	Parameters to only selected grid cells with an elevation higher than elev_min or lower than elev_max (default is '`NA`).
`elev_range`	Parameters discard the grid cell with a high elevational range (default is `NA`).
`year_min, year_max`	The oldest and youngest occurrences accepted (default is 1900-2021).
`nodate`	A boolean to accept occurrences without a date (can overlap with occurrences with a date; default `TRUE`).
`type_of_obs`	The type of observation to use in the study. 1: human observation, 2: observation, 3: preserved specimen, 4: living specimem, 5: fossil specimen, 6: material sample, 7: machine observation, 8: literature, 9: unknown (Default `c(1,2,3,8,9)`)
`df`	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
`x`	The name, age or depth of the rows of df (the samples).
`x.name`	A string describing the x axis (e.g. 'Sample Name', 'Age', 'Depth').
`minGridCells`	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
`weightedPresences`	A boolean to indicate whether the presence records should be weighted. Default is `FALSE`.
`bin_width`	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.
`shape`	The imposed shape of the species `pdfs`. We recommend using 'normal' for temperature variables and 'lognormal' for the variables that can only take positive values, such as precipitation or aridity. Default is 'normal' for all.

`npoints`	The number of points to be used to fit the `pdfs`. Default 200.
`geoWeighting`	A boolean to indicate if the species should be weighting by the squareroot of their extension when estimating a genus/family level taxon-climate relationships.
`climateSpaceWeighting`	A boolean to indicate if the species `pdfs` should be corrected for the modern distribution of the climate space (default `TRUE`).
`selectedTaxa`	A data frame assigns which taxa should be used for each variable (1 if the taxon should be used, 0 otherwise). The colnames should be the climate variables' names and the rownames the taxa names. Default is 1 for all taxa and all variables.
`distributions`	A dataframe containing the presence records of the studied proxies and their associated climate values.
`presenceThreshold`	All values above that threshold will be used in the reconstruction (e.g. if set at 1, all percentages below 1 will be set to 0 and the associated presences discarded). Default is 0.
`taxWeight`	One value among the following: 'originalData', 'presence/absence', 'percentages' or 'normalisation' (default).
`uncertainties`	A (vector of) threshold value(s) indicating the error bars that should be calculated (default both 50 and 95% ranges).

Value

A `crestObj` object that is used to store data and information for reconstructing climate

See Also

`https://gbif.github.io/parsers/apidocs/org/gbif/api/vocabulary/BasisOfRecord.html` for a detailed explanation of the types of observation.

`crop` *Crop the dataset obtained from* `crest.get_modern_data`

Description

Crop the dataset obtained from `crest.get_modern_data` according to an object of the class `SpatialPolygonsDataFram`

Usage

```
crop(x, shp)
```

Arguments

`x`	A `crestObj` produced by the `crest.get_modern_data` function.
`shp`	A shapefile to crop the data. Data points will be kept if their centroid is within the shape.

Examples

```
data(M1)
## We want only the data covering Nigeria
M2 <- M1[M1$COUNTRY == 'Nigeria', ]
data(reconstr)
reconstr.cropped <- crop(reconstr, M2)
data1 <- raster::rasterFromXYZ(reconstr$modelling$climate_space[, 1:3], crs=raster::crs(M1))
data2 <- raster::rasterFromXYZ(reconstr.cropped$modelling$climate_space[, 1:3], crs=raster::crs(M1))
layout(matrix(c(1,2,3,4), byrow=FALSE, ncol=2), width=1, height=c(0.2, 0.8))
plot_map_eqearth(data1, brks.pos=seq(13,29,2), colour_scale=TRUE,
                 title='Full dataset', zlim=c(13, 29))
plot_map_eqearth(data2, brks.pos=seq(13,29,2), colour_scale=TRUE,
                 title='Cropped dataset', zlim=c(13, 29))
```

`dbRequest` *Connect to the gbif4crest database*

Description

Connect to the gbif4crest_02 database by accessing the server on Amazon.

Usage

```
dbRequest(request, dbname = "gbif4crest_02")
```

Arguments

`request`	A SQL request to be executed.
`dbname`	The name of the data source database.

Value

The result of the request.

Examples

```
# Extracting the number of taxa recorded in the database
dbRequest("SELECT count(*) FROM taxa")

# Extracting all the taxa that have at least one occurrence in South Africa.
## Not run:
  southAfricaTaxa <- dbRequest(paste0(
    "SELECT DISTINCT taxa.* ",
    "FROM taxa, distrib_qdgc, geo_qdgc ",
    "WHERE taxa.taxonid=distrib_qdgc.taxonid ",
    "AND   distrib_qdgc.latitude=geo_qdgc.latitude ",
    "AND   distrib_qdgc.longitude=geo_qdgc.longitude ",
    "AND geo_qdgc.countryname='South Africa'"
  ))
  head(southAfricaTaxa)

## End(Not run)
```

`eqearth_get_ext` *Calculates the extent of the plot in the equal earth projection.*

Description

Calculates the extent of the plot in the equal earth projection.

Usage

```
eqearth_get_ext(ext, npoints = 15)
```

Arguments

`ext`	A set of coordinates.
`npoints`	The number of points used to draw the polygon along each dimension.

Value

The set of coordinates ext projected in equal earth.

Examples

```
eqearth_get_ext(c(-15, 50, 30, 70))
```

`excludeTaxa` *Excludes the list of taxa from the reconstructions.*

Description

Excludes the list of taxa from the reconstructions.

Usage

```
excludeTaxa(x, taxa, climate)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`taxa`	A vector of taxa to exclude.
`climate`	A vector of climate variables to unlink the taxa with.

Examples

```
data(reconstr)
print(reconstr$inputs$selectedTaxa)
reconstr <- excludeTaxa(reconstr, 'Taxon3', 'bio1')
## 'Taxon3' is now excluded from the reconstruction of 'bio1'.
print(reconstr$inputs$selectedTaxa)
```

explore_calibration_dataset

Extract distributions from the database

Description

This function will extract the distributions of all the studied climate proxy and plot the data on a map.

Usage

```
explore_calibration_dataset(
  taxaType,
  save = FALSE,
  filename = "calibrationDataset.pdf",
  col = viridis::viridis(22)[3:22],
  width = 7.48,
  height = 7.48,
  as.png = FALSE,
  png.res = 300,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  continents = NA,
  countries = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  dbname = "gbif4crest_02"
)
```

Arguments

`taxaType`	A numerical index (between 1 and 6) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents. The example dataset uses taxaType=0 (pseudo-data). Default is 1.
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'Reconstruction_climate.pdf'`.
`col`	A colour gradient.
`width, height`	The dimensions of the pdf file (default 7.48in ~19cm).
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`xmn`	The coordinates defining the study area.

`xmx`	The coordinates defining the study area.
`ymn`	The coordinates defining the study area.
`ymx`	The coordinates defining the study area.
`continents`	A vector of the continent names defining the study area.
`countries`	A vector of the country names defining the study area.
`realms`	A vector of the studied botanical realms defining the study area.
`biomes`	A vector of the studied botanical biomes defining the study area.
`ecoregions`	A vector of the studied botanical ecoregions defining the study area.
`dbname`	The name of the database. Default is `'gbif4crest_02'`.

Value

The distribution data

Examples

```
## Not run:
  d = explore_calibration_dataset(2, xmn=-85, xmx=-30, ymn=-60, ymx=15,
                                  save=TRUE, width = 4, height = 7.5)
  head(d)

## End(Not run)
```

`export` *Export the results*

Description

Export the results generated by the reconstruction

Usage

```
export(
  x,
  dataname = x$misc$site_info$site_name,
  climate = x$parameters$climate,
  loc = getwd(),
  as.csv = FALSE,
  fullPosterior = FALSE,
  loo = FALSE,
  weights = FALSE,
  pdfs = FALSE
)
```

Arguments

`x`	The name, age or depth of the rows of df (the samples).
`dataname`	The name of the site (default: `x$misc$site_info$site_name`)
`climate`	The climate data to export. Data for all climate variables are saved by default.
`loc`	The path where to export the data (default: working directory)
`as.csv`	Boolean to indicate if the data should be exported as csv (`TRUE`) or xlsx (`FALSE`, default)
`fullPosterior`	A boolean to export the climate posterior probability (default `FALSE`)
`loo`	A boolean to export the leave-one-out data if they exist (default `FALSE`)
`weights`	A boolean to export the weights derived from the percentages (default `FALSE`)
`pdfs`	A boolean to export the taxa's `pdfs` (default `FALSE`)

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    site_info = c(7.5, 7.5),
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example",
    leave_one_out = TRUE
  )
  export(reconstr, dataname='crest_example',
         fullPosterior=TRUE, weights=TRUE, loo=TRUE, pdfs=TRUE)

## End(Not run)
```

`export_pdfs` *Export the pdfs fitted for the different taxa.*

Description

Export the pdfs fitted for the different taxa.

Usage

```
export_pdfs(
  x,
  dataname = x$misc$site_info$site_name,
  climate = x$parameters$climate,
  taxa = x$inputs$taxa.name,
  loc = getwd(),
  as.csv = FALSE
)
```

Arguments

`x`	The name, age or depth of the rows of df (the samples).
`dataname`	The name of the site (default: `x$misc$site_info$site_name`)
`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`taxa`	The names of the taxa of interest. All the pdfs are saved by default.
`loc`	The path where to export the data (default: working directory)
`as.csv`	Boolean to indicate if the data should be exported as csv (`TRUE`) or xlsx (`FALSE`, default)

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    site_info = c(7.5, 7.5),
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example",
    leave_one_out = TRUE
  )
  export_pdfs(reconstr, dataname='crest_example')

## End(Not run)
```

`find.original.name` *Returns the name of the function argument in the global environment*

Description

Returns the name of the function argument in the global environment

Usage

```
find.original.name(x)
```

Arguments

`x`	The function argument

fit_pdfsp — *Fit the species* `pdfs`.

Description

Fit the species `pdfs`.

Usage

```
fit_pdfsp(climate, ccs, bin_width, shape, xrange, use_ccs = TRUE)
```

Arguments

`climate`	A vector of climatic values where the species is present.
`ccs`	A `ccs` object returned by `calib_clim_space`.
`bin_width`	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.
`shape`	The imposed shape of the species `pdfs`. We recommend using 'normal' for temperature variables and 'lognormal' for the variables that can only take positive values, such as precipitation or aridity. Default is 'normal' for all.
`xrange`	The climate gradient upon which the `pdf` with be defined.
`use_ccs`	Boolean to indicate if the `pdfsp` should be corrected by the distributin of the modern climate space

Value

The pdf of the species.

Examples

```
# Creating one randomised species
climate_species <- round(stats::rnorm(50, 15, 2), 1)

# Creating one randomised climate space
climate_space <- base::sample(0:300 / 10, 4000, replace = TRUE)

ccs <- calib_clim_space(climate_space, 2)
xrange <- fit_xrange(ccs, "normal", 2)
pdfsp <- fit_pdfsp(climate_species, ccs, 2, "normal", xrange)
plot(xrange, pdfsp, type = "l")

# Testing that the area under the curve is equal to 1.
all.equal(sum(pdfsp * (xrange[2] - xrange[1])), 1)
```

fit_xrange — *Define the climate gradient to fit the* pdfs.

Description

Define the climate gradient to fit the pdfs.

Usage

```
fit_xrange(ccs, shape, bin_width, npoints = 500)
```

Arguments

ccs	A ccs object returned by calib_clim_space.
shape	The imposed shape of the species pdfs. We recommend using 'normal' for temperature variables and 'lognormal' for the variables that can only take positive values, such as precipitation or aridity. Default is 'normal' for all.
bin_width	The width of the bins used to correct for unbalanced climate state. Use values that split the studied climate gradient in 15-25 classes (e.g. 2°C for temperature variables). Default is 1.
npoints	The number of points to be used to fit the pdfs. Default 200.

Value

A regularly spaced climate gradient with npoints points.

Examples

```
# Creating one randomised climate space
climate_space <- sample(0:300 / 10, 4000, replace = TRUE)
ccs <- calib_clim_space(climate_space, 2)
xrange <- fit_xrange(ccs, "normal", 2)
head(xrange)
```

get_taxa_type — *Returns the taxa type corresponding to the index.*

Description

Returns the taxa type corresponding to the index.

Usage

```
get_taxa_type(taxaType)
```

Arguments

taxaType	An integer between 0 and 6

`getClimateSpace` *Extract the distribution of the studied climate gradient(s) across the study area.*

Description

Extract the distribution of the studied climate gradient(s) across the study area.

Usage

```
getClimateSpace(
  climate,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  continents = NA,
  countries = NA,
  basins = NA,
  sectors = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  dbname = "gbif4crest_02"
)
```

Arguments

`climate`	A vector of the climate variables to extract. See `accClimateVariables` for the list of accepted values.
`xmn`	The coordinates defining the study area.
`xmx`	The coordinates defining the study area.
`ymn`	The coordinates defining the study area.
`ymx`	The coordinates defining the study area.
`continents`	A vector of the continent names defining the study area.
`countries`	A vector of the country names defining the study area.
`basins`	A vector of the ocean names defining the study area.
`sectors`	A vector of the marine sector names defining the study area.
`realms`	A vector of the studied botanical realms defining the study area.
`biomes`	A vector of the studied botanical biomes defining the study area.
`ecoregions`	A vector of the studied botanical ecoregions defining the study area.
`dbname`	The name of the data source database.

Value

A matrix of occurrence records with the associated climate.

See Also

`accClimateVariables` for a list of accepted climate variable names, `accCountryNames` for a list of accepted continent and country names, `accBasinNames` for a list of accepted basin and sector names, `accRealmNames` for a list of accepted realm, biome and ecoregion names.

Examples

```
climate <- getClimateSpace("bio1", -90, 90, -90, 90,
  continents = "Europe",
  countries = c("Germany", "Netherlands", "Sweden"),
  realms = "Palaearctic"
)
head(climate)
raster::plot(raster::rasterFromXYZ(climate), asp=1)
```

`getDistribTaxa` *Extract taxonID(s) corresponding to the taxonomic description*

Description

Extract all possible TaxonIDs corresponding to the provided taxonomical description, which can be at the family, the genus or the species levels.

Usage

```
getDistribTaxa(
  taxIDs,
  climate = NA,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  continents = NA,
  countries = NA,
  basins = NA,
  sectors = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  elev_min = NA,
  elev_max = NA,
  elev_range = NA,
  year_min = 1900,
  year_max = 2021,
  nodate = TRUE,
  type_of_obs = c(1, 2, 3, 8, 9),
  dbname = "gbif4crest_02"
)
```

Arguments

taxIDs	A vector of accepted Taxa IDs (as returned by getTaxonID).
climate	A vector of the climate variables to extract. See accClimateVariables for the list of accepted values.
xmn	The coordinates defining the study area.
xmx	The coordinates defining the study area.
ymn	The coordinates defining the study area.
ymx	The coordinates defining the study area.
continents	A vector of the continent names defining the study area.
countries	A vector of the country names defining the study area.
basins	A vector of the ocean names defining the study area.
sectors	A vector of the marine sector names defining the study area.
realms	A vector of the studied botanical realms defining the study area.
biomes	A vector of the studied botanical biomes defining the study area.
ecoregions	A vector of the studied botanical ecoregions defining the study area.
elev_min	Parameters to only selected grid cells with an elevation higher than elev_min or lower than elev_max (default is 'NA).
elev_max	Parameters to only selected grid cells with an elevation higher than elev_min or lower than elev_max (default is 'NA).
elev_range	Parameters discard the grid cell with a high elevational range (default is NA).
year_min	The oldest and youngest occurrences accepted (default is 1900-2021).
year_max	The oldest and youngest occurrences accepted (default is 1900-2021).
nodate	A boolean to accept occurrences without a date (can overlap with occurrences with a date; default TRUE).
type_of_obs	The type of observation to use in the study. 1: human observation, 2: observation, 3: preserved specimen, 4: living specimem, 5: fossil specimen, 6: material sample, 7: machine observation, 8: literature, 9: unknown (Default c(1,2,3,8,9))
dbname	The name of the data source database.

Value

A matrix of occurrence records with the associated climate.

See Also

getTaxonID for taxIDs, accClimateVariables for a list of accepted climate variable names, accCountryNames for a list of accepted continent and country names, accRealmNames for a list of accepted realm, biome and ecoregion names.

Examples

```
taxIDs <- getTaxonID("Zamiaceae", "Ceratozamia")
distrib <- getDistribTaxa(taxIDs, "bio1", -90, 90, -90, 90,
  continents = "Europe",
  countries = c("Germany", "Netherlands", "Sweden"),
  realms = "Palaearctic"
)
distrib
```

getSpeciesdiversity — *Calculates how many species compose the respose of each taxon.*

Description

Calculates how many species compose the respose of each taxon.

Usage

```
getSpeciesdiversity(x)
```

Arguments

x	A crestObj generated by the crest.reconstruct, loo or crest functions.

Examples

```
data(reconstr)
getSpeciesdiversity(reconstr)
```

getTaxonID — *Extract taxonID(s) corresponding to the taxonomic description*

Description

Extract all possible TaxonIDs corresponding to the provided taxonomical description, which can be at the family, the genus or the species levels.

Usage

```
getTaxonID(
  family = "",
  genus = "",
  species = "",
  taxaType = 1,
  dbname = "gbif4crest_02"
)
```

Arguments

family	The name of the family.
genus	The name of the genus.
species	The name of the species.
taxaType	A numerical index (between 1 and 6) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents. The example dataset uses taxaType=0 (pseudo-data). Default is 1.
dbname	The name of the data source database.

Value

A vector of unique taxonIDs.

Examples

```
getTaxonID("Zamiaceae")
getTaxonID("Zamiaceae", "Ceratozamia")
## \code{taxaType = 2} searches for beetles and not plants, so the next line returns nothing.
getTaxonID("Zamiaceae", "Ceratozamia", taxaType = 2)
```

`getTaxonomy` *Extract taxonID(s) corresponding to the taxonomic description*

Description

Extract all possible TaxonIDs corresponding to the provided taxonomical description, which can be at the family, the genus or the species levels.

Usage

```
getTaxonomy(
  family = "",
  genus = "",
  species = "",
  taxaType = 1,
  depth.out = 8,
  dbname = "gbif4crest_02"
)
```

Arguments

`family`	The name of the family.
`genus`	The name of the genus.
`species`	The name of the species.
`taxaType`	A numerical index (between 1 and 5) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for chironomids, 4 for foraminifers, 5 for diatoms and 6 for rodents.
`depth.out`	The taxonomic resolution of the output table. 1 for Kingdom, 2 for phylum, 3 for class_name, 4 for order_name, 5 for family, 6 for genus, 7 for species and 8 to also include the taxonID.
`dbname`	The name of the data source database.

Value

A vector of unique taxonIDs.

Examples

```
getTaxonomy("Zamiaceae    ")
getTaxonomy(genus="Ceratozamia", depth.out=6)
## \code{taxaType = 2} searches for beetles and not plants, so the next line returns nothing.
getTaxonomy("Zamiaceae", "Ceratozamia", taxaType = 2)
```

`includeTaxa` *Includes the list of taxa into the reconstructions.*

Description

Includes the list of taxa into the reconstructions.

Usage

```
includeTaxa(x, taxa, climate)
```

Arguments

`x`	A `crestObj` produced by one of the `crest`, `crest.get_modern_data`, `crest.calibrate`, `crest.reconstruct` or `loo` functions.
`taxa`	A vector of taxa to include.
`climate`	A vector of climate variables to link the taxa with.

Examples

```
data(reconstr)
print(reconstr$inputs$selectedTaxa)
reconstr <- includeTaxa(reconstr, reconstr$inputs$taxa.name, 'bio12')
## All the taxa are not selected for 'bio12', except for 'Taxon7' for which
## data are unavailable.
print(reconstr$inputs$selectedTaxa)
```

`isColourStr` *Test if* `R` *can interpret a string as a colour*

Description

Test if `R` can interpret a string as a colour

Usage

```
isColourStr(col)
```

Arguments

`col`	The string to be tested.

Value

A boolean value, TRUE if col is a valid colour, FALSE otherwise

Examples

```
isColourStr('black')
isColourStr('blakc')
```

---

loo *Performs the leave-one-out analysis*

Description

Repeat the repetation by removing one taxon at a time.

Usage

```
loo(x, verbose = TRUE)
```

Arguments

x	a crestObj produced by the crest.reconstruct or crest functions.
verbose	A boolean to print non-essential comments on the terminal (default TRUE).

Value

A crestObj object containing the reconstructions and all the associated data.

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- loo(reconstr)

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
lapply(reconstr$reconstructions$bio12$loo, head)
plot_loo(reconstr)
```

M1 *A shapefile of the world's country borders.*

Description

A shapefile of the world's country borders.

Usage

```
M1
```

Format

An object of class `SpatialPolygonsDataFrame` with 252 rows and 1 columns.

makeTransparent *Wrapper function of to add transparency to a colour.*

Description

Add transparency to the selected colours.

Usage

```
makeTransparent(colour, alpha)
```

Arguments

`colour`	A R colour
`alpha`	A value between 0 and 1 that defines the transparency 0 for full transparency and 1 for no transparency

Examples

```
makeTransparent('black',0.5)
makeTransparent('black',1:10/10)
makeTransparent(rainbow(10), 1:10/10)
```

`meanPositiveValues` *Calculate the mean of all stricly positive values.*

Description

Calculate the mean of all stricly positive values.

Usage

```
meanPositiveValues(x)
```

Arguments

`x`	A vector of values.

Value

The average of all the positive values. Returns `NaN` is no stricly positive values are found.

Examples

```
meanPositiveValues(-10:10)
```

`normalise` *Convert data into presence/absence data.*

Description

Convert data into presence/absence data.

Usage

```
normalise(df, col2convert = 2:ncol(df))
```

Arguments

`df`	The dataframe containing the data to convert.
`col2convert`	A vector of the columns to convert. Default is all the columns but the first, which contains an age, a depth or a sampleID.

Value

A vector of unique taxonIDs.

Examples

```
df <- data.frame(matrix(1:25, ncol = 5))
colnames(df) <- paste(rep("col", 5), 1:5, sep = "")
normalise(df)
normalise(df, col2convert = 3:5)
```

`plot.crestObj` *Plot the reconstructions.*

Description

Plot the reconstructions and their uncertainties if they exist.

Usage

```
## S3 method for class 'crestObj'
plot(
  x,
  climate = x$parameters$climate,
  uncertainties = x$parameters$uncertainties,
  optima = TRUE,
  add_modern = FALSE,
  simplify = FALSE,
  xlim = NA,
  ylim = NA,
  pt.cex = 0.8,
  pt.lwd = 0.8,
  pt.col = ifelse(simplify, "black", "white"),
  save = FALSE,
  width = 5.51,
  height = 5.51,
  as.png = FALSE,
  png.res = 300,
  filename = "Reconstruction.pdf",
  col = viridis::viridis(125)[26:125],
  ...
)
```

Arguments

`x`	A `crestObj` produced by either the `crest.reconstruct` or `crest`) functions.
`climate`	The climate variables to plot (default is all the reconstructed variables from x)
`uncertainties`	A (vector of) threshold value(s) indicating the error bars that should be calculated (default are the values stored in x).
`optima`	A boolean to indicate whether to plot the optimum (`TRUE`) or the mean (`FALSE`) estimates.
`add_modern`	Adds the modern climate values to the plot.
`simplify`	A boolean to indicate if the full distribution of uncertainties should be plooted (`FALSE`, default) or if they should be simplified to the uncertainty range(s).
`xlim`	the x limits (x1, x2) of the plot. Note that `x1 > x2` is allowed and leads to a ‘reversed axis’. The default value, `NULL`, indicates that the range of the finite values to be plotted should be used.
`ylim`	the y limits of the plot.

`pt.cex`	The size of the points (default 0.8).
`pt.lwd`	The thickness of the lines (default 0.8).
`pt.col`	The colour of the points and lines.
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`width, height`	The dimensions of the pdf file (default 5.51in ~14cm).
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'Reconstruction_climate.pdf'`.
`col`	A colour gradient.
`...`	other graphical parameters (see `par` and section 'Details' below).

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- loo(reconstr)

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot(reconstr)
plot(reconstr, climate='bio1', simplify = TRUE)
```

`plot_climateSpace` *Plot the studied climate space.*

Description

Plot the studied climate space.

Usage

```
plot_climateSpace(
  x,
  climate = x$parameters$climate,
  save = FALSE,
  filename = "Climate_space.pdf",
  as.png = FALSE,
  png.res = 300,
  width = 7.48,
  height = min(9, 3.5 * length(climate)),
  y0 = 0.5,
  add_modern = FALSE,
  resol = 0.25
)
```

Arguments

`x`	A `crestObj` generated by either the `crest.calibrate`, `crest.reconstruct` or `crest` functions.
`climate`	Climate variables to be used to generate the plot. By default all the variables are included.
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file (default `FALSE`).
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'Climate_space.pdf'`.
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`width`	The width of the output file in inches (default 7.48in ~ 19cm).
`height`	The height of the output file in inches (default 3in ~ 7.6cm per variables).
`y0`	The space to allocate to each title (default 0.3in ~ 0.76 cm.
`add_modern`	A boolean to add the location and the modern climate values to the plot (default `FALSE`).
`resol`	For advanced users only: if higher resolution data are used to estimate the `pdfs`, use this parameter to define the resolution of the maps maps on the figures. (default is 0.25 degrees to match with the default database).

Examples

```
## Not run:
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest.get_modern_data(
    pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- crest.calibrate(reconstr,
    geoWeighting = TRUE, climateSpaceWeighting = TRUE,
    bin_width = c(2, 20), shape = c("normal", "lognormal")
  )

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot_climateSpace(reconstr)
```

plot_combinedPDFs — *Plot representing how the* `pdfs` *combine to produce the reconstruction.*

Description

Plot representing how the `pdfs` combine to produce the reconstruction.

Usage

```
plot_combinedPDFs(
  x,
  samples = 1:length(x$inputs$x),
  climate = x$parameters$climate[1],
  optima = TRUE,
  xlim = NA,
  only.present = FALSE,
  only.selected = FALSE,
  col = crestr::colour_theme(1),
  save = FALSE,
  filename = "samplePDFs.pdf",
  as.png = FALSE,
  png.res = 300,
  width = 7.48,
  height = 5
)
```

Arguments

`x`	A `crestObj` generated by the `crest.reconstruct` or `crest` functions.
`samples`	The list of samples for which the plot should be plotted. All samples will be plotted by default.
`climate`	The climate variable to use to plot the variable. Default is first variable (`x$parameters$climate\[1\`
`optima`	A boolean to indicate whether to plot the optimum (`TRUE`) or the mean (`FALSE`) estimates.
`xlim`	The climate range to plot the pdfs on. Default is the full range used to fit the `pdfs` (x$modelling$xrange).
`only.present`	A boolean to only add the names of the taxa recorded in the sample (default `FALSE`).
`only.selected`	A boolean to only add the names of the selected taxa (default `FALSE`).
`col`	A range of colour values to colour the `pdfs`. Colours will be recycled to match the number of taxa.
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'samplePDFs.pdf'`.
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.

png.res	The resolution of the png file (default 300 pixels per inch).
width	The width of the output file in inches (default 7.48in ~ 19cm).
height	The height of the output file in inches (default 5in ~ 12.7cm).

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example",
    leave_one_out = FALSE
   )

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot_combinedPDFs(reconstr, samples=1:4, climate='bio12')
```

---

plot_diagram *Plot stratigraphic data as polygons or barplots.*

Description

This function plots stratigraphic data either as polygons or bars.

Usage

```
plot_diagram(
  x,
  bars = FALSE,
  col = "black",
  amplif = 5,
  save = FALSE,
  filename = "Diagram.pdf",
  width = 3.54,
  height = 9,
  as.png = FALSE,
  png.res = 300,
  yax_incr = 5,
  bar_width = 1,
  xlim = NA,
  tickAtSample = TRUE,
  col_pos = "black",
  col_neg = "grey80",
  title = NA
)
```

Arguments

`x`	A data frame of the data to plot (first column with age or depth) and the taxa in the following columns. x can also be a `crestObj`.
`bars`	A boolean that indicates if the data should be plotted as polygons (default: `bars=FALSE`) or vertical bars (`bars=TRUE`).
`col`	Colours to be used for the polygons. If the number of colours does not match the number of taxa, colors will be recyled.
`amplif`	A factor the show exageration on the diagram. Only for polygon plot. Default 5.
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'Diagram.pdf'`.
`width`	The width of the output file in inches (default 3.54in ~ 9cm).
`height`	The height of the output file in inches (default 9in ~ 23cm).
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`yax_incr`	Graphical parameters describing the increment size on the y-axis (default 5).
`bar_width`	Width of the bars of the barplot (default 1).
`xlim`	The range covered by the x-axis. Canbe adjusted to get round numbers on the x-ais. If smaller than the range overed by the data, the data will be truncated (default: range of the data).
`tickAtSample`	Boolean that indicates whether a tick mark should be added on the x-axis at the location of each sample (default `TRUE`).
`col_pos`	Graphical parameter for the barplot. Colour of all the positive values (default black).
`col_neg`	Graphical parameter for the barplot. Colour of all the negative values (default light grey).
`title`	Name to be added on top of the plot (default `NA`).

Examples

```
data(crest_ex)
plot_diagram(crest_ex, bars=TRUE, col='black', bar_width=0.8)
plot_diagram(crest_ex,  col=1:7, tickAtSample=FALSE)
## Not run:
  plot_diagram(crest_ex, save=TRUE, filename='testDiagram.pdf',
               bars=TRUE, col_pos='cornflowerblue', col_neg='darkgreen',
               bar_width=0.8, xlim=c(3,15))

## End(Not run)
```

plot_loo *Plot de results of the leave-one-out analysis.*

Description

Plot de results of the leave-one-out analysis.

Usage

```
plot_loo(
  x,
  optima = TRUE,
  climate = x$parameters$climate,
  taxanames = x$inputs$taxa.name,
  save = FALSE,
  filename = "Diagram_loo.pdf",
  as.png = FALSE,
  png.res = 300,
  width = 3.54,
  height = 9,
  yax_incr = NA,
  bar_width = 1,
  xlim = NA,
  tickAtSample = FALSE,
  col_pos = "black",
  col_neg = "grey80",
  title = NA
)
```

Arguments

`x`	A data frame of the data to plot (first column with age or depth) and the taxa in the following columns. `x` can also be a `crestObj`.
`optima`	A boolean to indicate whether to plot the optimum (`TRUE`) or the mean (`FALSE`) estimates.
`climate`	Climate variables to be used to generate the plot. By default all the variables are included.
`taxanames`	A list of taxa to use for the plot (default is all the recorded taxa).
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'Diagram_loo_climate.pdf'`.
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`width`	The width of the output file in inches (default 3.54in ~ 9cm).
`height`	The height of the output file in inches (default 9in ~ 23cm).

`yax_incr`	Graphical parameters describing the increment size on the y-axis (default 5).
`bar_width`	Width of the bars of the barplot (default 1).
`xlim`	The range covered by the x-axis. Canbe adjusted to get round numbers on the x-ais. If smaller than the range overed by the data, the data will be truncated (default: range of the data).
`tickAtSample`	Boolean that indicates whether a tick mark should be added on the x-axis at the location of each sample (default `TRUE`).
`col_pos`	Graphical parameter for the barplot. Colour of all the positive values (default black).
`col_neg`	Graphical parameter for the barplot. Colour of all the negative values (default grey80).
`title`	Name to be added on top of the plot (default `NA`).

Examples

```
## Not run:
  data(crest_ex)
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest(
    df = crest_ex, pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"), bin_width = c(2, 20),
    shape = c("normal", "lognormal"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- loo(reconstr)

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot_loo(reconstr, yax_incr=c(0.5, 50), bar_width=0.8,
         col_pos=c('blue','cornflowerblue'), col_neg=c('red', 'goldenrod3'))
```

`plot_map_eqearth` *Plots raster data in equal earth projection.*

Description

Plots raster data in equal earth projection.

Usage

```
plot_map_eqearth(
  dat,
  ext = raster::extent(dat),
  zlim = range(raster::values(dat), na.rm = TRUE),
  col = viridis::viridis(20),
  brks.pos = c(0, 1),
  brks.lab = brks.pos,
  npoints = 15,
  nlines = 9,
  title = "",
  colour_scale = TRUE,
  top_layer = NA,
  top_layer.col = "ghostwhite",
  site_xy = NA,
  dim = NA,
  scale = 1
)
```

Arguments

`dat`	The raster data to plot.
`ext`	The extent to use to plot the data. (default is extent of dat)
`zlim`	The range of the values to plot. (default is estimated from dat)
`col`	The color gradient to use. (default is viridis)
`brks.pos`	The position where to draw tick marks on the legend
`brks.lab`	The labels to add where the tickmarks are draw (default is tickmarks position)
`npoints`	The number of points used to draw the polygons and lines along each dimension. (default is 15 for a smooth result)
`nlines`	The number of coordinate lines to add in the background (default is 9)
`title`	A description title (default is empty).
`colour_scale`	A boolean to add the colour scale to the plot (default `TRUE`).
`top_layer`	A raster to overlay on top of the map (e.g. a distribution).
`top_layer.col`	A colour for plotting top_layer (default 'ghostwhite').
`site_xy`	Coordinates of a location to add to the plot.
`dim`	The dimension of the plotting window in inches (default dev.size()).
`scale`	A scaling parameter to correct for the change of font size created by layout().

`plot_taxaCharacteristics` *Plot the distribution and responses of the studied taxa*

Description

Plot the distribution and responses of the studied taxa

Usage

```
plot_taxaCharacteristics(
  x,
  taxanames = x$inputs$taxa.name,
  climate = x$parameters$climate,
  col.density = viridis::plasma(20),
  col.climate = viridis::viridis(22)[3:20],
  save = FALSE,
  filename = "taxaCharacteristics.pdf",
  as.png = FALSE,
  png.res = 300,
  width = 7.48,
  w0 = 0.3,
  height = 3 * length(climate),
  h0 = 0.4,
  add_modern = FALSE,
  resol = 0.25
)
```

Arguments

`x`	A `crestObj` generated by either the `crest.calibrate`, `crest.reconstruct`, `loo` or `crest` functions.
`taxanames`	A list of taxa to use for the plot (default is all the recorded taxa).
`climate`	Climate variables to be used to generate the plot. By default all the variables are included.
`col.density`	The colour gradient to use to map the density of species (topleft map).
`col.climate`	The colour gradient to use to map the climate gradients (left column).
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file. Default is `FALSE`.
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'taxaCharacteristics.pdf'`.
`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).
`width`	The width of the output file in inches (default 7.48in ~ 19cm).
`w0`	The width of the left column with the names.
`height`	The height of the output file in inches (default 3in ~ 7.6cm per variables).
`h0`	The vertical space used for the x-axes.
`add_modern`	A boolean to add the location and the modern climate values to the plot (default `FALSE`).
`resol`	For advanced users only: if higher resolution data are used to estimate the `pdfs`, use this parameter to define the resolution of the maps on the figures. (default is 0.25 degrees to match with the default database)

Examples

```
## Not run:
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest.get_modern_data(
    pse = crest_ex_pse, taxaType = 0, df = crest_ex,
    climate = c("bio1", "bio12"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- crest.calibrate(reconstr,
    geoWeighting = TRUE, climateSpaceWeighting = TRUE,
    bin_width = c(2, 20), shape = c("normal", "lognormal")
  )

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot_taxaCharacteristics(reconstr, taxanames='Taxon1')
```

`plot_violinPDFs` *Plot the pdfs as violins*

Description

Plot the pdfs as violins

Usage

```
plot_violinPDFs(
  x,
  climate = x$parameters$climate[1],
  taxanames = x$input$taxa.name,
  col = viridis::viridis(20),
  ylim = range(x$modelling$xrange[[climate]]),
  save = FALSE,
  filename = "violinPDFs.pdf",
  width = 7.48,
  height = 5,
  as.png = FALSE,
  png.res = 300
)
```

Arguments

`x`	A `crestObj` generated by either the `crest.calibrate`, `crest.reconstruct` or `crest` functions.
`climate`	Climate variables to be used to generate the plot. By default all the variables are included.
`taxanames`	A list of taxa to use for the plot (default is all the recorded taxa).
`col`	A vector of colours that will be linearly interpolated to give a unique colour to each taxon.
`ylim`	The climate range to plot the pdfs on. Default is the full range used to fit the `pdfs` (x$modelling$xrange)
`save`	A boolean to indicate if the diagram shoud be saved as a pdf file (default `FALSE`).
`filename`	An absolute or relative path that indicates where the diagram should be saved. Also used to specify the name of the file. Default: the file is saved in the working directory under the name `'violinPDFs.pdf'`.
`width`	The width of the output file in inches (default 7.48in ~ 19cm).
`height`	The height of the output file in inches (default 3in ~ 7.6cm per variables).

`as.png`	A boolean to indicate if the output should be saved as a png. Default is `FALSE` and the figure is saved as a pdf file.
`png.res`	The resolution of the png file (default 300 pixels per inch).

Examples

```
## Not run:
  data(crest_ex_pse)
  data(crest_ex_selection)
  reconstr <- crest.get_modern_data(
    pse = crest_ex_pse, taxaType = 0,
    climate = c("bio1", "bio12"),
    selectedTaxa = crest_ex_selection, dbname = "crest_example"
  )
  reconstr <- crest.calibrate(reconstr,
    geoWeighting = TRUE, climateSpaceWeighting = TRUE,
    bin_width = c(2, 20), shape = c("normal", "lognormal")
  )

## End(Not run)
## example using pre-saved reconstruction obtained with the previous command.
data(reconstr)
plot_violinPDFs(reconstr, save=FALSE, ylim=c(5,35),
   taxanames=c(reconstr$inputs$taxa.name[c(2,4,5,1)], 'Taxon'),
   col=c('darkblue', 'firebrick3'))
```

`reconstr` *A `crestObj` ran with the example dataset.*

Description

A `crestObj` ran with the example dataset. Useful to illustrate many functions of the package.

Usage

```
reconstr
```

Format

An object of class `crestObj` of length 5.

Index