

# Package ‘crestr’

January 8, 2025

**Title** A Probabilistic Approach to Reconstruct Past Climates Using Palaeoecological Datasets

**Version** 1.3.1.9000

**Description** Applies 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. More information can be obtained from Chevalier (2022) <[doi:10.5194/cp-18-821-2022](https://doi.org/10.5194/cp-18-821-2022)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.1

**Imports** clipr,  
DBI,  
data.table,  
methods,  
openxlsx,  
plot3D,  
plyr,  
RPostgres,  
RSQLite,  
scales,  
stringr,  
terra,  
viridis

**Suggests** knitr,  
rmarkdown,  
pals

**Depends** R (>= 2.10)

**Language** en-us

**VignetteBuilder** knitr

**URL** <https://github.com/mchevalier2/crestr>, <https://www.manuelchevalier.com/crestr/index.html>

**BugReports** <https://github.com/mchevalier2/crestr/issues>

## Contents

accBasinNames . . . . .	3
accClimateVariables . . . . .	4
accCountryNames . . . . .	4
accRealmNames . . . . .	5
calib_clim_space . . . . .	6
check_coordinates . . . . .	6
cite_climate_data . . . . .	7
cite_crest . . . . .	7
cite_distrib_data . . . . .	8
cite_method . . . . .	8
climate_from_xy . . . . .	9
close_db_connection . . . . .	9
colour_theme . . . . .	10
connect_local_sqlite3 . . . . .	11
connect_online . . . . .	11
convert2percentages . . . . .	12
convert2presenceAbsence . . . . .	13
copy_crest . . . . .	13
createPSE . . . . .	14
crest . . . . .	15
crest.calibrate . . . . .	17
crest.get_modern_data . . . . .	19
crest.reconstruct . . . . .	21
crest.set_modern_data . . . . .	22
crest.simplify . . . . .	24
crestObj . . . . .	24
crest_ex . . . . .	27
crest_ex_pse . . . . .	28
crest_ex_selection . . . . .	28
crop . . . . .	29
dbDownload . . . . .	29
dbRequest . . . . .	30
dbSubset . . . . .	31
eqearth_get_ext . . . . .	32
excludeTaxa . . . . .	33
explore_calibration_dataset . . . . .	33
export . . . . .	35
export_pdfs . . . . .	36
find.original.name . . . . .	37
fit_pdfsp . . . . .	38
fit_xrange . . . . .	39
f_locid . . . . .	40
getClimateSpace . . . . .	40
getDistribTaxa . . . . .	42
getResol . . . . .	43
getSpeciesdiversity . . . . .	44
getTaxaTypeFromTaxID . . . . .	44
getTaxonID . . . . .	45
getTaxonomy . . . . .	46
get_taxa_type . . . . .	47

includeTaxa . . . . .	47
is.crestObj . . . . .	48
isColourStr . . . . .	48
loo . . . . .	49
M1 . . . . .	50
makeTransparent . . . . .	50
meanPositiveValues . . . . .	51
normalise . . . . .	51
pdf_ranges . . . . .	52
plot.crestObj . . . . .	52
plot_climateSpace . . . . .	54
plot_combinedPDFs . . . . .	56
plot_diagram . . . . .	58
plot_loo . . . . .	59
plot_map_eqearth . . . . .	61
plot_scatterPDFs . . . . .	62
plot_taxaCharacteristics . . . . .	64
plot_violinPDFs . . . . .	65
PSE_log . . . . .	67
reconstr . . . . .	67
taxonComposition . . . . .	68
testConnection . . . . .	68

## Index 69

---

accBasinNames	<i>Return the list of oceans and seas.</i>
---------------	--

---

### 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 accepted names.

### Value

A list of accepted names.

### See Also

<https://www.marineregions.org/downloads.php>

### 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.

### See Also

<https://www.worldclim.org/data/bioclim.html> for details on the 'bio' data, <https://csidotinfo.wordpress.com/2019/01/24/global-aridity-index-and-potential-evapotranspiration-climate-database-for-details-on-ai/>, <https://www.ncei.noaa.gov/products/world-ocean-atlas> for details on the sea surface temperature, sea surface salinity and nutrient Concentration data, or <https://psl.noaa.gov/data/gridded/data.noaa.oisst.v2.highres.html> for the sea ice concentration data.

### Examples

```
## Not run:
  accClimateVariables()
  accClimateVariables(v='bio12')

## End(Not run)
```

---

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.

**See Also**

<https://www.naturalearthdata.com/downloads/10m-cultural-vectors/10m-admin-0-countries/>

**Examples**

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

---

accRealmNames	<i>Return the list of the realms and associated biomes and ecoregions.</i>
---------------	--

---

**Description**

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

**Usage**

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

**Arguments**

realms      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.

**See Also**

<https://www.worldwildlife.org/publications/terrestrial-ecoregions-of-the-world>

**Examples**

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

---

calib_clim_space	<i>Calibrate the distribution of the modern climate space.</i>
------------------	--

---

**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	<i>Check if the coordinates are correct.</i>
-------------------	--

---

**Description**

Check if the coordinates are correct.

**Usage**

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

**Arguments**

xmn, xmx, ymn, ymx	The coordinates defining the study area.
--------------------	--

**Value**

Return a set of valid coordinates.

**Examples**

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

---

<code>cite_climate_data</code>	<i>Returns the references associated with the climate data used to fit the pdfs.</i>
--------------------------------	--

---

### Description

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

### Usage

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

### Arguments

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

### Value

A list of references to add if the data generated by crestr are published.

---

<code>cite_crest</code>	<i>Returns the list of references associated to the reconstruction.</i>
-------------------------	---

---

### Description

Returns the list of references associated to the reconstruction.

### Usage

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

### Arguments

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

### Value

A list of references to add if the data generated by crestr are published.

---

cite_distrib_data	<i>Returns the references associated with the GBIF data used to fit the pdfs.</i>
-------------------	---

---

### Description

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

### Usage

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

### Arguments

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

### Value

A list of references to add if the data generated by crestr are published.

---

cite_method	<i>Returns the references associated with the development of CREST.</i>
-------------	---

---

### Description

Returns the references associated with the development of CREST.

### Usage

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

### Arguments

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

### Value

A list of references to add if the data generated by crestr are published.



---

climate_from_xy	<i>Extract The climate values associated to a set of coordinates.</i>
-----------------	---

---

### 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

```
## Not run:  
climate_from_xy(50, 10, c('bio1', 'ai'))  
climate_from_xy(50, 10)  
  
## End(Not run)
```

---

close_db_connection	<i>Disconnect the database connection.</i>
---------------------	--

---

### Description

Disconnect the database connection.

### Usage

```
close_db_connection(db)
```

**Arguments**

db                      An active database connection

**Value**

No return value, function called to close the connection to the database.

**Examples**

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

---

colour_theme	<i>Returns a vector of colours</i>
--------------	------------------------------------

---

**Description**

Returns a vector of colours

**Usage**

```
colour_theme(n)
```

**Arguments**

n                      An index to select the colour theme

**Value**

A vector of colours.

**Examples**

```
colour_theme(1)
```

---

connect_local_sqlite3	<i>Connect to the gbif4crest calibration database</i>
-----------------------	---

---

**Description**

Connect to the gbif4crest\_02 database using a local SQLite3 copy.

**Usage**

```
connect_local_sqlite3(dbname = "gbif4crest_02.sqlite3")
```

**Arguments**

dbname	The complete path to the SQLite3 file. The name should end by '.sqlite3'
--------	--

**Value**

An active connection to a database

**See Also**

The SQLite3 database can be downloaded from [https://figshare.com/articles/dataset/GBIF\\_for\\_CREST\\_database/6743207](https://figshare.com/articles/dataset/GBIF_for_CREST_database/6743207).

**Examples**

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

---

connect_online	<i>Connect to the gbif4crest calibration database</i>
----------------	---

---

**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.cvqgy2mnjwgtg.eu-west-3.rds.amazonaws.com'.
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	<i>Convert abundance data into percentage data.</i>
---------------------	---

---

**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 $\geq$ 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 <code>crestObj</code> produced by the <code>crest.reconstruct</code> or <code>crest</code> functions.
climate	A vector of the climate variables to extract. See <code>accClimateVariables</code> 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.

**Value**

No return value. This function is called to copy the crest data 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

*Creates a spreadsheet with the format required for a PSE.*

---

**Description**

Creates a spreadsheet with the format required for a PSE from a list of taxa.

**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 <code>proxy_species_equivalency.xlsx</code> .

**Value**

No return value, called to create a PSE file.

## Examples

```
data(crest_ex)
createPSE(taxa = colnames(crest_ex)[-1],
          loc=file.path(tempdir(), 'pse.xlsx')
)
```

---

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,
  climateSpaceWeighting.type = "linear",
  presenceThreshold = 0,
  taxWeight = "normalisation",
  uncertainties = c(0.5, 0.95),
  leave_one_out = FALSE,
  verbose = TRUE,
  dbname = "gbif4crest_02"
)
```

## Arguments

<code>df</code>	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
<code>climate</code>	A vector of the climate variables to extract. See <a href="#">accClimateVariables</a> for the list of accepted values.
<code>pse</code>	A pollen-Species equivalency table. See <a href="#">createPSE</a> for details.
<code>taxaType</code>	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 <code>taxaType=0</code> (pseudo-data). Default is 1.
<code>distributions</code>	A dataframe containing the presence records of the studied proxies and their associated climate values.
<code>site_info</code>	A vector containing the coordinates of the study site. Default <code>c(NA, NA)</code> .
<code>site_name</code>	The name of the dataset (default NA).
<code>xmn, xmx, ymn, ymx</code>	The coordinates defining the study area.
<code>continents</code>	A vector of the continent names defining the study area.
<code>countries</code>	A vector of the country names defining the study area.
<code>realms</code>	A vector of the studied botanical realms defining the study area.
<code>biomes</code>	A vector of the studied botanical biomes defining the study area.
<code>ecoregions</code>	A vector of the studied botanical ecoregions defining the study area.
<code>minGridCells</code>	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
<code>selectedTaxa</code>	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.
<code>bin_width</code>	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.
<code>shape</code>	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.
<code>npoints</code>	The number of points to be used to fit the pdfs. Default 200.
<code>ai.sqrt</code>	A boolean to indicate whether ai values should be square-root transformed (default FALSE).
<code>geoWeighting</code>	A boolean to indicate if the species should be weighting by the square root of their extension when estimating a genus/family level taxon-climate relationships.
<code>climateSpaceWeighting</code>	A boolean to indicate if the species pdfs should be corrected for the modern distribution of the climate space (default TRUE).
<code>climateSpaceWeighting.type</code>	A correction factor for the climate space weighting correction to limit the edge effects. Either 'linear' (default), 'sqrt' or 'log'.



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

```
## 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), 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)

## End(Not run)
```

---

crest.calibrate	<i>Fit the species and proxy pdfs</i>
-----------------	---------------------------------------

---

### 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,
climateSpaceWeighting.type = x$parameters$climateSpaceWeighting.type,
verbose = TRUE
)

```

## Arguments

x	A <a href="#">crestObj</a> produced by the <a href="#">crest.get_modern_data</a> 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 square root 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).
climateSpaceWeighting.type	A correction factor for the climate space weighting correction to limit the edge effects. Either 'linear' (default), 'sqrt' or 'log'.
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

```

## Not run:
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
)

## End(Not run)

```

---

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,
  climateWithObs = FALSE,
  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 <a href="#">createPSE</a> 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 <a href="#">accClimateVariables</a> for the list of accepted values.

<code>df</code>	A data frame containing the data to reconstruct (counts, percentages or presence/absence data).
<code>ai.sqrt</code>	A boolean to indicate whether ai values should be square-root transformed (default FALSE).
<code>xmn, xmx, ymn, ymx</code>	The coordinates defining the study area.
<code>continents</code>	A vector of the continent names defining the study area.
<code>countries</code>	A vector of the country names defining the study area.
<code>basins</code>	A vector of the ocean names defining the study area.
<code>sectors</code>	A vector of the marine sector names defining the study area.
<code>realms</code>	A vector of the studied botanical realms defining the study area.
<code>biomes</code>	A vector of the studied botanical biomes defining the study area.
<code>ecoregions</code>	A vector of the studied botanical ecoregions defining the study area.
<code>minGridCells</code>	The minimum number of unique presence data necessary to estimate a species' climate response. Default is 20.
<code>climateWithObs</code>	A boolean to indicate whether all climate values from the calibration dataset be included (FALSE, default) or only the climate values that corresponds to proxy observations (TRUE). Only useful in if the climate space weighting is activated.
<code>elev_min, elev_max</code>	Parameters to only selected grid cells with an elevation higher than <code>elev_min</code> or lower than <code>elev_max</code> (default is 'NA').
<code>elev_range</code>	Parameters discard the grid cell with a high elevation range (default is NA).
<code>year_min, year_max</code>	The oldest and youngest occurrences accepted (default is 1900-2021).
<code>nodate</code>	A boolean to accept occurrences without a date (can overlap with occurrences with a date; default TRUE).
<code>type_of_obs</code>	The type of observation to use in the study. 1: human observation, 2: observation, 3: preserved specimen, 4: living specimen, 5: fossil specimen, 6: material sample, 7: machine observation, 8: literature, 9: unknown (Default <code>c(1, 2, 3, 8, 9)</code> )
<code>selectedTaxa</code>	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.
<code>site_info</code>	A vector containing the coordinates of the study site. Default <code>c(NA, NA)</code> .
<code>site_name</code>	The name of the dataset (default NA).
<code>dbname</code>	The name of the database. Default is 'gbif4crest_02' and data will be extracted from the online database. The SQLite3 version of the database can also be used here by providing the complete path to a file ending by <code>.sqlite3</code> , e.g. <code>/path/to/file/gbif4crest_02.sqlite3</code>
<code>verbose</code>	A boolean to print non-essential comments on the terminal (default TRUE).

### Value

A `crestObj` object containing the spatial distributions.

**See Also**

The SQLite3 database can be downloaded from [https://figshare.com/articles/dataset/GBIF\\_for\\_CREST\\_database/6743207](https://figshare.com/articles/dataset/GBIF_for_CREST_database/6743207).

**Examples**

```
## Not run:
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$modeling$distributions, head)

## End(Not run)
```

---

crest.reconstruct	<i>Reconstruct climate from fossil data</i>
-------------------	---

---

**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 <code>crestObj</code> produced by the <code>crest.calibrate</code> 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 leave 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)
## Not run:
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)

## End(Not run)
```

---

`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 = 20,
  selectedTaxa = NA,
  site_info = c(NA, NA),
  site_name = NA,
  site_climate = rep(NA, length(climate)),
  verbose = TRUE
)
```



---

crest.simplify	<i>Simplify a crestObj into a dataframe.</i>
----------------	--

---

### Description

Simplify a crestObj with reconstructed values into a dataframe.

### Usage

```
crest.simplify(x, optima = TRUE)
```

### Arguments

x	A <a href="#">crestObj</a> produced by the <a href="#">crest</a> , <a href="#">crest.reconstruct</a> or <a href="#">loo</a> functions.
optima	A boolean to indicate whether to plot the optimum (TRUE) or the mean (FALSE) estimates.

### Value

A dataframe with the age/depth of each sample and all the best reconstructed values.

### Examples

```
head(crest.simplify(reconstr))
```

---

crestObj	<i>Create a crestObj object.</i>
----------	----------------------------------

---

### 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,
climateSpaceWeighting.type = "linear",
climateWithObs = FALSE,
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 <a href="#">accClimateVariables</a> for the list of accepted values.
pse	A pollen-Species equivalency table. See <a href="#">createPSE</a> 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 elevation 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 specimen, 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 square root 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).
climateSpaceWeighting.type	A correction factor for the climate space weighting correction to limit the edge effects. Either 'linear' (default), 'sqrt' or 'log'.
climateWithObs	A boolean to indicate whether all climate values from the calibration dataset be included (FALSE, default) or only the climate values that corresponds to proxy observations (TRUE). Only useful in if the climate space weighting is activated.
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

See vignette('technicalities') for details about the structure of the object. See also <https://gbif.github.io/parsers/apidocs/org/gbif/api/vocabulary/BasisOfRecord.html> for a detailed explanation of the types of observation.

---

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 integer 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 variables.*


---

### 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)

---

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 <code>crestObj</code> produced by the <code>crest.get_modern_data</code> function.
shp	A shapefile ( <code>spatVect</code> ) to crop the data. Data points will be kept if their centroid is within the shape.

## Value

An cropped version of the input `crestObj`.

## Examples

```
## Not run:
data(M1)
M1 <- terra::unwrap(M1)
## We want only the data covering Nigeria
M2 <- M1[M1$COUNTRY == 'Nigeria', ]
data(reconstr)
reconstr.cropped <- crop(reconstr, M2)
data1 <- terra::rast(reconstr$modelling$climate_space[, 1:3],
                     crs=terra::crs(M1), type='xyz')
data2 <- terra::rast(reconstr.cropped$modelling$climate_space[, 1:3],
                     crs=terra::crs(M1), type='xyz')
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))

## End(Not run)
```

---

dbDownload

---

*Download the `gbif4crest_02` dataset from figShare.*


---

## Description

Download the `gbif4crest_02` dataset from figShare.

**Usage**

```
dbDownload(
  filename = "gbif4crest_02.zip",
  lite = TRUE,
  res = "15min",
  timeout = 10000
)
```

**Arguments**

filename	The path and name of the file where the database should be saved.
lite	A boolean (default TRUE) to indicate if the full database should be downloaded (including the raw presences from GBIF; <code>lite = FALSE</code> ) or only the curated data ( <code>lite = TRUE</code> ).
res	The spatial resolution of the dataset. Either 15min or 5min.
timeout	Maximum duration in seconds of the download. If the file is not fully downloaded after timeout seconds, it will be interrupted.

**See Also**

The full SQLite3 database can be downloaded from [https://figshare.com/articles/dataset/GBIF\\_for\\_CREST\\_database/6743207](https://figshare.com/articles/dataset/GBIF_for_CREST_database/6743207).

**Examples**

```
## Not run:
dbDownload() ## This will download the database in your working directory.

## End(Not run)
```

---

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

```
## Not run:
# 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.
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)
```

---

dbSubset

---

*Create a subset of the global calibration dataset*


---

## Description

Create a subset of the global calibration dataset

## Usage

```
dbSubset(
  taxaType,
  xmn = NA,
  xmx = NA,
  ymn = NA,
  ymx = NA,
  dbname = "gbif4crest_02",
  out = "gbif4crest_reduced",
  verbose = TRUE
)
```

## 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.
xmn, xmx, ymn, ymx	The coordinates defining the study area.
dbname	The name of the database. Default is 'gbif4crest_02' and data will be extracted from the online database. The SQLite3 version of the database can also be used here by providing the complete path to a file ending by .sqlite3, e.g. /path/to/file/gbif4crest_02.sqlite3
out	The name or path of the new dataset
verbose	A boolean to print non-essential comments on the terminal (default TRUE).

**See Also**

The full SQLite3 database can be downloaded from [https://figshare.com/articles/dataset/GBIF\\_for\\_CREST\\_database/6743207](https://figshare.com/articles/dataset/GBIF_for_CREST_database/6743207).

**Examples**

```
## Not run:
dbSubset(2, xmn=0, xmx=15, ymn=0, ymx=15, out='example.sqlite3')

## End(Not run)
```

---

eqearth_get_ext	<i>Calculates the extent of the plot in the equal earth projection.</i>
-----------------	---

---

**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**

```
## Not run:
eqearth_get_ext(c(-15, 50, 30, 70))

## End(Not run)
```



---

excludeTaxa	<i>Excludes the list of taxa from the reconstructions.</i>
-------------	--

---

### Description

Excludes the list of taxa from the reconstructions.

### Usage

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

### Arguments

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

### Value

Return the updated `crestObj`.

### 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	<i>Extract distributions from the database</i>
-----------------------------	--

---

### 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 should 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, xmx, ymn, 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:
#> Replace 'tempdir()' by the location where to save the sample (e.g. 'getwd()')
d = explore_calibration_dataset(2, xmn=-85, xmx=-30, ymn=-60, ymx=15,
                               save=TRUE, width = 4, height = 7.5,
                               filename=file.path(tempdir(), 'calibrationDataset.pdf'))
)
head(d)

```

```
## End(Not run)
```

---

export	<i>Export the results</i>
--------	---------------------------

---

## 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,
  fullUncertainties = FALSE,
  loo = FALSE,
  weights = FALSE,
  pdfs = FALSE
)
```

## Arguments

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

## Value

No return value, function called to export the results.

## 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
)
#> Replace 'tempdir()' by the location where the sample should be saved (e.g. 'getwd()')
export(reconstr, dataname='crest_example',
  fullUncertainties=TRUE, weights=TRUE, loo=TRUE, pdfs=TRUE,
  loc=tempdir())

## 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 <a href="#">accClimateVariables</a> 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)

**Value**

No return value, function called to export the PDFs as files.

**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
)
#> Replace 'tempdir()' by the location where the sample should be saved (e.g. 'getwd()')
export_pdfs(reconstr,
             dataname='crest_example',
             loc=tempdir()
)

## End(Not run)
```

---

find.original.name	<i>Returns the name of the function argument in the global environment</i>
--------------------	--

---

**Description**

Returns the name of the function argument in the global environment

**Usage**

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

**Arguments**

x	The function argument
---	-----------------------

**Value**

The name of the function argument in the global environment.

---

fit_pdfsp	<i>Fit the species pdfs.</i>
-----------	------------------------------

---

## Description

Fit the species pdfs.

## Usage

```
fit_pdfsp(
  climate,
  ccs,
  bin_width,
  shape,
  xrange,
  use_ccs = TRUE,
  climateSpaceWeighting.type = "linear"
)
```

## Arguments

climate	A vector of climatic values where the species is present.
ccs	A ccs object returned by <a href="#">calib_clim_space</a> .
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 will be defined.
use_ccs	Boolean to indicate if the pdfsp should be corrected by the distribution of the modern climate space
climateSpaceWeighting.type	A correction factor for the climate space weighting correction to limit the edge effects. Either 'linear' (default), 'sqrt' or 'log'.

## 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 <a href="#">calib_clim_space</a> .
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)
```

---

f_locid	<i>Calculates a unique identifier from coordinates.</i>
---------	---

---

### Description

Calculates a unique identifier from coordinates.

### Usage

```
f_locid(long, lat, resol)
```

### Arguments

long	The longitude of the grid cell of interest
lat	The latitude of the grid cell of interest
resol	The spatial resolution of the database to target (1/12 for gbif4crest_03)

### Value

A unique integer describing the cell.

### Examples

```
f_locid(0, 0, 1/12)
```

---

getClimateSpace	<i>Extract the distribution of the studied climate gradient(s) across the study area.</i>
-----------------	---

---

### 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,
```



```

    elev_min = NA,
    elev_max = NA,
    elev_range = NA,
    dbname = "gbif4crest_02"
  )

```

### Arguments

climate	A vector of the climate variables to extract. See <a href="#">accClimateVariables</a> for the list of accepted values.
xmn, xmx, ymn, 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, 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 elevation range (default is NA).
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

```

## Not run:
climate <- getClimateSpace("bio1", -90, 90, -90, 90,
  continents = "Europe",
  countries = c("Germany", "Netherlands", "Sweden"),
  realms = "Palearctic"
)
head(climate)
terra::plot(terra::rast(climate, type='xyz'), asp=1)

## End(Not run)

```

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, 7, 8),
  dbname = "gbif4crest_02"
)
```

**Arguments**

taxIDs	A vector of accepted Taxa IDs (as returned by <a href="#">getTaxonID</a> ).
climate	A vector of the climate variables to extract. See <a href="#">accClimateVariables</a> for the list of accepted values.
xmn, xmx, ymn, 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, 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 elevation 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 specimen, 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**

```
## Not run:
taxIDs <- getTaxonID("Zamiaceae", "Ceratozamia")
distrib <- getDistribTaxa(taxIDs, "bio1", -90, 90, -90, 90,
  continents = "Europe",
  countries = c("Germany", "Netherlands", "Sweden"),
  realms = "Palearctic"
)
distrib

## End(Not run)
```

---

getResol

---

*Calculates a unique identifier from coordinates.*


---

**Description**

Calculates a unique identifier from coordinates.

**Usage**

```
getResol(crest)
```

**Arguments**

crest                    A crestObj object.

**Value**

A unique integer describing the cell.

**Examples**

```
getResol(reconstr)
```

---

```
getSpeciesdiversity
```

*Calculates how many species compose the response of each taxon.*


---

**Description**

Calculates how many species compose the response of each taxon.

**Usage**

```
getSpeciesdiversity(x)
```

**Arguments**

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

**Value**

Return the number of composing species of each taxon.

**Examples**

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

---

```
getTaxaTypeFromTaxID
```

*Returns the taxa type corresponding to the taxID.*


---

**Description**

Returns the taxa type corresponding to the taxID.

**Usage**

```
getTaxaTypeFromTaxID(taxID)
```

**Arguments**

**taxID** An integer between 0 and 6

**Value**

Returns the taxa type ID corresponding to the taxon ID.

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**

```
## Not run:
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)

## End(Not run)
```

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",
  crest = NA
)
```

**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.
crest	A crestObj to be used to refine the selection to a specific study area. Set to NA by default (global search).

**Value**

A vector of unique taxonIDs.

**Examples**

```
## Not run:
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)

## End(Not run)
```

---

get_taxa_type	<i>Returns the taxa type corresponding to the index.</i>
---------------	--

---

**Description**

Returns the taxa type corresponding to the index.

**Usage**

```
get_taxa_type(taxaType)
```

**Arguments**

taxaType	An integer between 0 and 6
----------	----------------------------

**Value**

Returns the taxa type corresponding to the index.

---

includeTaxa	<i>Includes the list of taxa into the reconstructions.</i>
-------------	--

---

**Description**

Includes the list of taxa into the reconstructions.

**Usage**

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

**Arguments**

x	A <a href="#">crestObj</a> produced by one of the <a href="#">crest</a> , <a href="#">crest.get_modern_data</a> , <a href="#">crest.calibrate</a> , <a href="#">crest.reconstruct</a> or <a href="#">loo</a> functions.
taxa	A vector of taxa to include.
climate	A vector of climate variables to link the taxa with.

**Value**

Return the updated [crestObj](#).

**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)
```

---

is.crestObj	<i>Test if x is a crestObj.</i>
-------------	---------------------------------

---

**Description**

Test if x is a crestObj.

**Usage**

```
is.crestObj(x)
```

**Arguments**

x	The object to be tested
---	-------------------------

**Value**

TRUE (x is a crestObj) or FALSE (not a crestObj).

---

isColourStr	<i>Test if R can interpret a string as a colour</i>
-------------	---

---

**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	<i>Performs the leave-one-out analysis</i>
-----	--

---

## Description

Repeat the reconstructions by removing one taxon at a time.

## Usage

```
loo(x, climate = x$parameters$climate, verbose = TRUE)
```

## Arguments

x	a <a href="#">crestObj</a> produced by the <a href="#">crest.reconstruct</a> or <a href="#">crest</a> functions.
climate	A vector of the climate variables to extract. See <a href="#">accClimateVariables</a> for the list of accepted values.
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	<i>A shapefile of the world's country borders. Use 'M1 &lt;- terra::unwrap(M1)' to render the object usable in R.</i>
----	---

---

**Description**

A shapefile of the world's country borders. Use 'M1 <- terra::unwrap(M1)' to render the object usable in R.

**Usage**

M1

**Format**

An object of class PackedSpatVector of length 1.

---

makeTransparent	<i>Wrapper function of to add transparency to a colour.</i>
-----------------	---

---

**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

**Value**

Return a colour with the provided level of transparency.

**Examples**

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

---

meanPositiveValues	<i>Calculate the mean of all strictly positive values.</i>
--------------------	--

---

**Description**

Calculate the mean of all strictly positive values.

**Usage**

```
meanPositiveValues(x)
```

**Arguments**

x                      A vector of values.

**Value**

The average of all the positive values. Returns NaN if no strictly positive values are found.

**Examples**

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

---

normalise	<i>Normalises the percentages</i>
-----------	-----------------------------------

---

**Description**

Normalises the percentages

**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)
```

---

pdf_ranges	<i>Calculate the climate tolerance of the taxa from their pdfs.</i>
------------	---

---

### Description

Calculate the climate tolerance of the taxa from their pdfs.

### Usage

```
pdf_ranges(
  x,
  climate = x$parameters$climate,
  taxanames = x$input$taxa.name,
  uncertainties = x$parameters$uncertainties,
  orderby = NA
)
```

### Arguments

x	A <a href="#">crestObj</a> generated by either the <a href="#">crest.calibrate</a> , <a href="#">crest.reconstruct</a> or <a href="#">crest</a> 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).
uncertainties	A (vector of) threshold value(s) indicating the error bars that should be calculated (default are the values stored in x).
orderby	A string ('name', or one of the climate variables) to sort the output table

### Value

The set of coordinates ext projected in equal earth.

### Examples

```
data(reconstr)
pdf_ranges(reconstr, climate='bio1')
pdf_ranges(reconstr, climate='bio12', orderby='bio1', uncertainties=c(0.2, 0.6, 0.95))
```

---

plot.crestObj	<i>Plot the reconstructions.</i>
---------------	----------------------------------

---

### Description

Plot the reconstructions and their uncertainties if they exist.

**Usage**

```
## S3 method for class 'crestObj'
plot(
  x,
  climate = x$parameters$climate[1],
  uncertainties = x$parameters$uncertainties,
  optima = TRUE,
  add_modern = FALSE,
  simplify = FALSE,
  as.anomaly = FALSE,
  anomaly.base = x$misc$site_info$climate[climate],
  xlim = NA,
  ylim = NA,
  pt.cex = 0.8,
  pt.lwd = 0.8,
  pt.col = ifelse(simplify, "black", "white"),
  col.hiatus = "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 <a href="#">crestObj</a> produced by the <a href="#">crest</a> , <a href="#">crest.reconstruct</a> or <a href="#">loo</a> 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 plotted (FALSE, default) or if they should be simplified to the uncertainty range(s).
as.anomaly	A boolean to indicate if the reconstructions should be plotted as absolute values (Default, FALSE) or anomalies '(TRUE).
anomaly.base	The anomaly value. Should be a vector with the same length as climate. Default values are the climate values corresponding to the location of the record (site_info in <a href="#">crest.get_modern_data</a> ).
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 <a href="#">finite</a> 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.
col.hiatus	A colour for the hiatus(es) of the record (default white)
save	A boolean to indicate if the diagram should 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 <a href="#">graphical parameters</a> (see <a href="#">par</a> and section 'Details' below).

### Value

No return value, this function is used to plot.

### 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, as.anomaly=TRUE)
```

---

plot_climateSpace	<i>Plot the studied climate space.</i>
-------------------	--

---

### Description

Plot the studied climate space.

**Usage**

```
plot_climateSpace(
  x,
  climate = x$parameters$climate,
  bin_width = x$parameters$bin_width[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 = 0.4,
  add_modern = FALSE,
  resol = getResol(x)
)
```

**Arguments**

x	A <a href="#">crestObj</a> generated by either the <a href="#">crest.calibrate</a> , <a href="#">crest.reconstruct</a> or <a href="#">crest</a> functions.
climate	Climate variables to be used to generate the plot. By default all the variables are included.
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.
save	A boolean to indicate if the diagram should 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 '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 0.4in ~ 1 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).

**Value**

No return value, this function is used to plot.

## 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")
)
plot_climateSpace(reconstr)

## End(Not run)
```

---

plot_combinedPDFs	<i>Plot representing how the pdfs combine to produce the reconstruction.</i>
-------------------	--

---

## Description

Plot representing how the pdfs combine to produce the reconstruction.

## Usage

```
plot_combinedPDFs(
  x,
  ages = range(x$inputs$x),
  samples = which(x$inputs$x >= min(ages) & x$inputs$x <= max(ages)),
  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 <code>crestObj</code> generated by the <code>crest.reconstruct</code> or <code>crest</code> functions.
ages	An age range to subset the samples to plot. By default, all the samples will be selected.
samples	The list of sample indexes 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 should 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).

### Value

No return value, this function is used to plot.

### 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, ages=c(6,9), climate='bio12')
```

---

plot_diagram	<i>Plot stratigraphic data as polygons or barplots.</i>
--------------	---

---

## 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 = diff(range(x$inputs$x))/length(x$inputs$x),
  xlim = NA,
  tickAtSample = TRUE,
  col_pos = "black",
  col_neg = "grey80",
  title = NA,
  src = 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 <a href="#">crestObj</a> .
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 recycled.
amplif	A factor the show exaggeration on the diagram. Only for polygon plot. Default 5.
save	A boolean to indicate if the diagram should 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 1/50th of the x range).
xlim	The range covered by the x-axis. Can be adjusted to get round numbers on the x-axis. If smaller than the range covered 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).
src	A graphical parameter used by the plot_loo() function.

### Value

No return value, this function is used to plot.

### 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)
#> Replace 'tempdir()' by the location where the sample should be saved (e.g. 'getwd()')
plot_diagram(crest_ex, save=TRUE,
             filename=file.path(tempdir(), 'testDiagram.pdf'),
             bars=TRUE, col_pos='cornflowerblue', col_neg='darkgreen',
             bar_width=0.8, xlim=c(3,15))
```

---

plot\_loo

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

---

### Description

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

### Usage

```
plot_loo(
  x,
  optima = TRUE,
  climate = x$parameters$climate[unlist(lapply(x$reconstructions, function(y)
    return("loo" %in% names(y))))],
  taxanames = x$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 = diff(range(x$inputs$x))/length(x$inputs$x),
xlim = NA,
tickAtSample = FALSE,
sort = NA,
filter = 0,
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 <a href="#">crestObj</a> .
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 should 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. Can be adjusted to get round numbers on the x-axis. If smaller than the range covered 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).
sort	A string to sort the order of the taxa from the highest to lowest anomalies (sort='incr') or from the lowest to highest (sort='decr'). Use the default value NA to keep the taxa unsorted.
filter	A threshold value that determines the mean absolute anomaly value required for the taxon to be plotted (default 0 means that all taxa are plotted)
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).

**Value**

When used with a crestObj, it returns the average leave-one-out values for each selected taxa

**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)
loo_vals <- 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	<i>Plots raster data in equal earth projection.</i>
------------------	---

---

**Description**

Plots raster data in equal earth projection.

**Usage**

```
plot_map_eqearth(
  dat,
  ext = as.vector(terra::ext(dat)),
  zlim = range(terra::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
)
```

**Arguments**

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

**Value**

The set of coordinates `ext` projected in equal earth.

---

<code>plot_scatterPDFs</code>	<i>Plot the pdf optima and uncertainty ranges in a climate biplot</i>
-------------------------------	---

---

**Description**

Plot the pdf optima and uncertainty ranges in a climate biplot

**Usage**

```
plot_scatterPDFs(
  x,
  climate = x$parameters$climate[1:2],
  taxanames = x$input$taxa.name,
  uncertainties = x$parameters$uncertainties,
  xlim = range(x$model$climate_space[, climate[1]]),
  ylim = range(x$model$climate_space[, climate[2]]),
  add_modern = FALSE,
  save = FALSE,
  filename = "scatterPDFs.pdf",
  width = 5.51,
  height = 5.51,
  as.png = FALSE,
  png.res = 300
)
```

**Arguments**

x	A <code>crestObj</code> generated by either the <code>crest.calibrate</code> , <code>crest.reconstruct</code> or <code>crest</code> functions.
climate	Names of the two climate variables to be used to generate the plot. By default plot. By default the first two variables are included.
taxanames	A list of taxa to use for the plot (default is all the recorded taxa).
uncertainties	A (vector of) threshold value(s) indicating the error bars that should be calculated (default are the values stored in x).
xlim	ylim The climate range to plot the data. Default is the full range of the observed climate space.
ylim	the y limits of the plot.
add_modern	A boolean to add the location and the modern climate values to the plot (default FALSE).
save	A boolean to indicate if the diagram should 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 '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).

**Value**

A table with the climate tolerances of all the taxa

**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)
dat <- plot_scatterPDFs(reconstr, save=FALSE,
  taxanames=c(reconstr$inputs$taxa.name[c(2,4,5,1)]))
dat
```

---

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)[5:20],
  col.climate = viridis::viridis(22)[3:20],
  save = FALSE,
  filename = "taxaCharacteristics.pdf",
  as.png = FALSE,
  png.res = 300,
  width = 7.48,
  w0 = 0.2,
  height = 3 * length(climate) + h0,
  h0 = 0.4,
  add_modern = FALSE,
  resol = getResol(x)
)
```

## Arguments

x	A <a href="#">crestObj</a> generated by either the <a href="#">crest.calibrate</a> , <a href="#">crest.reconstruct</a> , <a href="#">loo</a> or <a href="#">crest</a> 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 (top left map).
col.climate	The colour gradient to use to map the climate gradients (left column).
save	A boolean to indicate if the diagram should 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)

### Value

No return value, this function is used to plot.

### 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")
)
plot_taxaCharacteristics(reconstr, taxanames='Taxon1')

## End(Not run)
```

---

plot_violinPDFs	<i>Plot the pdfs as violins</i>
-----------------	---------------------------------

---

### 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 <code>crestObj</code> generated by either the <code>crest.calibrate</code> , <code>crest.reconstruct</code> or <code>crest</code> 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 ( <code>x\$modelling\$xrange</code> )
save	A boolean to indicate if the diagram should 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 '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).

**Value**

A table with the climate tolerances of all the taxa

**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)
ranges <- plot_violinPDFs(reconstr, save=FALSE, ylim=c(5,35),
  taxanames=c(reconstr$inputs$taxa.name[c(2,4,5,1)]),
  col=c('darkblue', 'firebrick3'))
lapply(ranges, head)
```

---

PSE_log	<i>Displays the log of the proxy-species association</i>
---------	--

---

### Description

Displays the log of the proxy-species association performed by the `crest.get_modern_data()` or `crest.set_modern_data()`

### Usage

`PSE_log(x)`

### Arguments

`x` A `crestObj` generated by one of the following functions: `crest.get_modern_data`, `crest.set_modern_data`, `crest.calibrate`, `crest.reconstruct`, `loo`, or `crest` functions.

### Examples

`PSE_log(reconstr)`

---

<code>reconstr</code>	<i>A <code>crestObj</code> ran with the example dataset.</i>
-----------------------	--

---

### 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.

---

taxonComposition	<i>Return the size of the distribution of each composing species of each taxon</i>
------------------	--

---

### Description

Return the size of the distribution of each composing species of each taxon

### Usage

```
taxonComposition(x, taxanames = x$input$taxa.name)
```

### Arguments

x	A <a href="#">crestObj</a> generated by either the <a href="#">crest.calibrate</a> , <a href="#">crest.reconstruct</a> or <a href="#">crest</a> functions.
taxanames	A list of taxa to use (default is all the recorded taxa).

### Value

A list with the number of unique occurrences for each composing species

### Examples

```
## Not run:
data(reconstr)
taxonComposition(reconstr)

## End(Not run)
```

---

testConnection	<i>Test if a connection can be established with the calibration data</i>
----------------	--

---

### Description

Test if a connection can be established with the calibration data

### Usage

```
testConnection(dbname = "gbif4crest_02")
```

### Arguments

dbname	The name of the data source database.
--------	---------------------------------------

### Value

TRUE if the connection can be established, else FALSE.

# Index

## \* datasets

- crest\_ex, [27](#)
- crest\_ex\_pse, [28](#)
- crest\_ex\_selection, [28](#)
- M1, [50](#)
- reconstr, [67](#)
- accBasinNames, [3, 41](#)
- accClimateVariables, [4, 14, 16, 19, 23, 25, 36, 41–43, 49](#)
- accCountryNames, [4, 41, 43](#)
- accRealmNames, [5, 41, 43](#)
- calib\_clim\_space, [6, 38, 39](#)
- check\_coordinates, [6](#)
- cite\_climate\_data, [7](#)
- cite\_crest, [7](#)
- cite\_distrib\_data, [8](#)
- cite\_method, [8](#)
- climate\_from\_xy, [9](#)
- close\_db\_connection, [9](#)
- colour\_theme, [10](#)
- connect\_local\_sqlite3, [11](#)
- connect\_online, [11](#)
- convert2percentages, [12](#)
- convert2presenceAbsence, [13](#)
- copy\_crest, [13](#)
- createPSE, [14, 16, 19, 25](#)
- crest, [7, 8, 14, 15, 24, 33, 44, 47, 49, 52, 53, 55, 56, 63, 64, 66–68](#)
- crest.calibrate, [7, 8, 17, 21, 23, 33, 47, 52, 55, 63, 64, 66–68](#)
- crest.get\_modern\_data, [7, 8, 18, 19, 29, 33, 47, 67](#)
- crest.reconstruct, [7, 8, 14, 21, 24, 33, 44, 47, 49, 52, 53, 55, 56, 63, 64, 66–68](#)
- crest.set\_modern\_data, [22, 67](#)
- crest.simplify, [24](#)
- crest\_ex, [27](#)
- crest\_ex\_pse, [28](#)
- crest\_ex\_selection, [28](#)
- crestObj, [7, 8, 14, 17, 18, 20–24, 24, 29, 33, 44, 47, 49, 52, 53, 55, 56, 58, 60, 63, 64, 66–68](#)
- crop, [29](#)
- dbDownload, [29](#)
- dbRequest, [30](#)
- dbSubset, [31](#)
- eqearth\_get\_ext, [32](#)
- excludeTaxa, [33](#)
- explore\_calibration\_dataset, [33](#)
- export, [35](#)
- export\_pdfs, [36](#)
- f\_locid, [40](#)
- find.original.name, [37](#)
- finite, [53](#)
- fit\_pdfsp, [6, 38](#)
- fit\_xrange, [39](#)
- get\_taxa\_type, [47](#)
- getClimateSpace, [40](#)
- getDistribTaxa, [42](#)
- getResol, [43](#)
- getSpeciesdiversity, [44](#)
- getTaxaTypeFromTaxID, [44](#)
- getTaxonID, [42, 43, 45](#)
- getTaxonomy, [46](#)
- graphical parameters, [54](#)
- includeTaxa, [47](#)
- is.crestObj, [48](#)
- isColourStr, [48](#)
- loo, [7, 8, 22, 24, 33, 44, 47, 49, 53, 64, 67](#)
- M1, [50](#)
- makeTransparent, [50](#)
- meanPositiveValues, [51](#)
- normalise, [51](#)
- par, [54](#)
- pdf\_ranges, [52](#)
- plot.crestObj, [52](#)
- plot\_climateSpace, [54](#)
- plot\_combinedPDFs, [56](#)

plot\_diagram, [58](#)  
plot\_loo, [59](#)  
plot\_map\_eqearth, [61](#)  
plot\_scatterPDFs, [62](#)  
plot\_taxaCharacteristics, [64](#)  
plot\_violinPDFs, [65](#)  
PSE\_log, [67](#)  
  
reconstr, [67](#)  
  
taxonComposition, [68](#)  
testConnection, [68](#)