

# Package ‘crestr’

October 23, 2020

**Title** What the Package Does (One Line, Title Case)

**Version** 0.0.0.9000

**Description** What the package does (one paragraph).

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.1

**Imports** DBI,  
RPostgreSQL,  
rio,  
graphics,  
plot3D,  
viridis,  
grDevices

**Depends** R (>= 2.10)

**Suggests** testthat,  
knitr,  
rmarkdown

**VignetteBuilder** knitr

## R topics documented:

accClimateVariables . . . . .	2
accContinentNames . . . . .	2
accRealmNames . . . . .	3
calib_clim_space . . . . .	3
close_db_connection . . . . .	4
connect_online . . . . .	4
convert2percentages . . . . .	5
convert2presenceAbsence . . . . .	6
crest . . . . .	6
crest.calibrate . . . . .	8
crest.get_modern_data . . . . .	9
crest.reconstruct . . . . .	10
crest_ex . . . . .	11

crest_ex_pse . . . . .	12
crest_ex_selection . . . . .	12
crestObj . . . . .	13
dbRequest . . . . .	14
fit_pdfsp . . . . .	15
fit_xrange . . . . .	16
getClimateSpace . . . . .	16
getDistribTaxa . . . . .	18
getTaxonID . . . . .	19
getTaxonomy . . . . .	20
meanPositiveValues . . . . .	21
normalise . . . . .	21

<b>Index</b>	<b>22</b>
--------------	-----------

---

accClimateVariables	<i>Describes all the variables available in the database.</i>
---------------------	---

---

### Description

Describes all the variables available in the database.

### Usage

```
accClimateVariables()
```

### Value

A data frame descriptive of the climate variables available in the database.

### Examples

```
accClimateVariables()
```

---

accContinentNames	<i>Return the list of the continents and associated countries.</i>
-------------------	--

---

### Description

Return the list of the continents and associated countries.

### Usage

```
accContinentNames(dbname = "gbif4crest_02")
```

### Arguments

dbname            The name of the database. Default is gbif4crest\_02.

### Value

A list with elements that correspond to the country names of each continent.

**Examples**

```
accContinentNames()
```

---

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(ecoregion = TRUE, dbname = "gbif4crest_02")
```

**Arguments**

ecoregion	A boolean to choose whether to get the ecoregions names.
dbname	The name of the database. Default is gbif4crest_02.

**Value**

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

**Examples**

```
accRealmNames()
```

---

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	All the climate values observed across the study area.
bin_width	The width of the climate bins.

**Value**

A ccs object that will be used by [fit\\_pdfsp](#).

**Examples**

```
# Extracting the number of taxa recorded in the database
calib_clim_space(sample(0:300 / 10, 4000, replace = TRUE), 2)
```

---

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
----	-------------------------------

**Examples**

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

## End(Not run)
```

---

connect_online	<i>Connect to the gbif4crest 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.cvqgy2mnjwgtg.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)
```

---

```
crest
```

*Connect to the gbif4crest database*

---

### Description

Connect to the gbif4crest\_02 database by accessing the server on Amazon.

### Usage

```
crest(
  df,
  pse,
  taxaType,
  climate,
  xmn = -180,
  xmx = 180,
  ymn = -90,
  ymx = 90,
  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,
geoWeighting = TRUE,
climateSpaceWeighting = TRUE,
presenceThreshold = 0,
taxWeight = "normalisation",
dbname = "gbif4crest_02"
)

```

### Arguments

df	.
pse	.
taxaType	.
climate	A vector of the climate variables to extract.
xmn	The coordinates defining the study area.
xmx	The coordinates defining the study area.
ymn	The coordinates defining the study area.
ymx	The coordinates defining the study area.
continents	A vector of the continent names defining the study area.
countries	A vector of the country names defining the study area.
realms	A vector of the studied botanical realms defining the study area.
biomes	A vector of the studied botanical biomes defining the study area.
ecoregions	A vector of the studied botanical ecoregions defining the study area.
minGridCells	.
selectedTaxa	.
bin_width	.
shape	.
npoints	The number of points to be used to fit the pdfs.
geoWeighting	The number of points to be used to fit the pdfs.
climateSpaceWeighting	The number of points to be used to fit the pdfs.
presenceThreshold	.
taxWeight	'originalData', 'presence/absence', 'percentages' or 'normalisation'
dbname	The name of the database. Default is gbif4crest_02.

### Value

The parameters to be used by crest()

## Examples

```
data(crest_ex)
data(crest_ex_pse)
data(crest_ex_selection)
recons <- 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"
)
plot(recons)
```

---

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 = rep(1, length(x$parameters$climate)),
  shape = rep("normal", length(x$parameters$climate)),
  npoints = 500,
  geoWeighting = TRUE,
  climateSpaceWeighting = TRUE
)
```

## Arguments

x	a crestObj produced by the crest.climate_space function.
bin_width	.
shape	.
npoints	The number of points to be used to fit the pdfs.
geoWeighting	The number of points to be used to fit the pdfs.
climateSpaceWeighting	The number of points to be used to fit the pdfs.

## Value

A crest() object containing the spatial distributions and the climate space



## Examples

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

## 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,
  taxa.name = unique(pse[, "ProxyName"]),
  xmn = -180,
  xmx = 180,
  ymn = -90,
  ymx = 90,
  continents = NA,
  countries = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  minGridCells = 20,
  selectedTaxa = NA,
  dbname = "gbif4crest_02"
)
```

## Arguments

pse	.
taxaType	.

climate	A vectof of the climate variables to extract.
taxa.name	.
xmn	The coordinates defining the study area.
xmx	The coordinates defining the study area.
ymn	The coordinates defining the study area.
ymx	The coordinates defining the study area.
continents	A vector of the continent names defining the study area.
countries	A vector of the country names defining the study area.
realms	A vector of the studied botanical realms defining the study area.
biomes	A vector of the studied botanical biomes defining the study area.
ecoregions	A vector of the studied botanical ecoregions defining the study area.
minGridCells	.
selectedTaxa	.
dbname	The name of the database. Default is gbif4crest_02.

### Value

A crest() object containing the spatial distributions

### Examples

```
## Not run:
data(crest_ex_pse)
data(crest_ex_selection)
x <- crest.get_modern_data(
  pse = crest_ex_pse, taxaType = 0,
  climate = c("bio1", "bio12"),
  selectedTaxa = crest_ex_selection, dbname = "crest_example"
)
x
lapply(x$modelling$distributions, head)

## End(Not run)
```

---

crest.reconstruct	<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.reconstruct(
  x,
  df,
  selectedTaxa = x$inputs$selectedTaxa,
  presenceThreshold = 0,
  taxWeight = "normalisation"
)
```

**Arguments**

**x** a crestObj produced by the crest.fit\_pdfs function.  
**df** .  
**selectedTaxa** .  
**presenceThreshold** .  
**taxWeight** 'originalData', 'presence/absence', 'percentages' or 'normalisation'

**Value**

A crest() object containing the reconstructions and all the associated data.

**Examples**

```

data(crest_ex)
data(crest_ex_pse)
data(crest_ex_selection)
x <- crest.get_modern_data(
  pse = crest_ex_pse, taxaType = 0,
  climate = c("bio1", "bio12"),
  taxa.name = colnames(crest_ex)[-1],
  selectedTaxa = crest_ex_selection, dbname = "crest_example"
)
x <- crest.calibrate(x,
  geoWeighting = TRUE, climateSpaceWeighting = TRUE,
  bin_width = c(2, 20), shape = c("normal", "lognormal")
)
x <- crest.reconstruct(x, crest_ex, selectedTaxa = crest_ex_selection)
plot(x)

```

---

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 7 columns (taxa):

**Age:** Age of each sample

**Taxon1:** Percentage of Taxon1 in each sample.

**Taxon2:** Percentage of Taxon2 in each sample.

**Taxon3:** Percentage of Taxon3 in each sample.

**Taxon4:** Percentage of Taxon4 in each sample.

**Taxon5:** Percentage of Taxon5 in each sample.

**Taxon6:** Percentage of Taxon6 in each sample.

**Taxon7:** Percentage of Taxon7 in each sample.

---

crest\_ex\_pse

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

---

### Description

A database indicating the taxonomy of the example proxies.

### Usage

crest\_ex\_pse

### Format

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

**Level:** An integr indicating the taxonomic resolution (1 family, 2 genus, 3 species, 4 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 which taxa should be used to reconstruct each climate variable.

### Usage

crest\_ex\_selection

### Format

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

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

**bio12:** The second variable to reconstruct (annual precipitation)

---

crestObj	Create a crest() object.
----------	--------------------------

---

## Description

Creates a crest() object with all default parameters.

## Usage

```
crestObj(
  taxa.name,
  pse,
  taxaType,
  climate,
  xmn,
  xmx,
  ymn,
  ymx,
  continents,
  countries,
  realms,
  biomes,
  ecoregions,
  df = NA,
  x = NA,
  x.name = "",
  minGridCells = 20,
  bin_width = rep(1, length(climate)),
  shape = rep("normal", length(climate)),
  npoints = 500,
  geoWeighting = TRUE,
  climateSpaceWeighting = TRUE,
  selectedTaxa = NA,
  presenceThreshold = 0,
  taxWeight = "normalisation"
)
```

## Arguments

taxa.name	.
pse	.
taxaType	.
climate	A vectof of the climate variables to extract.
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.
df	.
x	.
x.name	.
minGridCells	.
bin_width	.
shape	.
npoints	The number of points to be used to fit the pdfs.
geoWeighting	The number of points to be used to fit the pdfs.
climateSpaceWeighting	The number of points to be used to fit the pdfs.
selectedTaxa	.
presenceThreshold	.
taxWeight	'originalData', 'presence/absence', 'percentages' or 'normalisation'

**Value**

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

---

dbRequest	<i>Connect to the gbif4crest database</i>
-----------	---

---

**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 database. Default is gbif4crest_02.

**Value**

The result of the request.

**Examples**

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

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

## End(Not run)
```

fit\_pdfsp

*Fit the species pdfs.***Description**

Fit the species pdfs.

**Usage**

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

**Arguments**

climate	A vector of climatic values where the species is present.
ccs	A ccs object returned by <a href="#">calib_clim_space</a> .
bin_width	The width of the climate bins.
shape	The shape of the species pdfs. Use 'normal' or 'lognormal'.
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

**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.
sum(pdfsp * (xrange[2] - xrange[1])) == 1

```

---

fit_xrange	<i>Define teh climate gradient to fit the pdfs.</i>
------------	---

---

### Description

Define teh 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 shape of the species pdfs. Use 'normal' or 'lognormal'.
bin_width	The width of the climate bins.
npoints	The number of points to be used to fit the pdfs.

### 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)

```

---

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 = -180,  
  xmx = 180,  
  ymn = -90,  
  ymx = 90,  
  continents = NA,  
  countries = NA,  
  realms = NA,  
  biomes = NA,  
  ecoregions = NA,  
  dbname = "gbif4crest_02"  
)
```

**Arguments**

climate	A vector of the climate variables to extract.
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**

A matrix of occurrence records with the associated climate.

**See Also**

[accClimateVariables](#) for a list of accepted climate variable names, [accContinentNames](#) for a list of accepted continent and country names, [accRealmNames](#) for a list of accepted realm, biome and ecoregion names.

**Examples**

```
climate <- getClimateSpace("bio1", -90, 90, -90, 90,  
  continents = "Europe",  
  countries = c("Germany", "Netherlands", "Sweden"),  
  realms = "Palearctic"  
)  
head(climate)  
plot(climate[, -3], asp = 1)
```

---

getDistribTaxa

---

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


---

## Description

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

## Usage

```
getDistribTaxa(
  taxIDs,
  climate,
  xmn = -180,
  xmx = 180,
  ymn = -90,
  ymx = 90,
  continents = NA,
  countries = NA,
  realms = NA,
  biomes = NA,
  ecoregions = NA,
  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.
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

A matrix of occurrence records with the associated climate.

## See Also

[getTaxonID](#) for taxIDs, [accClimateVariables](#) for a list of accepted climate variable names, [accContinentNames](#) for a list of accepted continent and country names, [accRealmNames](#) for a list of accepted realm, biome and ecoregion names.

**Examples**

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

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 5) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for foraminifers, 4 for diatoms, 5 for chironomids and 6 for rodents.
dbname	The name of the database. Default is gbif4crest_02.

**Value**

A vector of unique taxonIDs.

**Examples**

```
getTaxonID("Zamiaceae")
getTaxonID("Zamiaceae", "Ceratozamia")
getTaxonID("Zamiaceae", "Ceratozamia", taxaType = 2)
```

---

getTaxonomy

---

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


---

## Description

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

## Usage

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

## Arguments

family	The name of the family.
genus	The name of the genus.
species	The name of the species.
taxaType	A numerical index (between 1 and 5) to define the type of palaeoproxy used: 1 for plants, 2 for beetles, 3 for foraminifers, 4 for diatoms, 5 for chironomids 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 database. Default is gbif4crest_02.

## Value

A vector of unique taxonIDs.

## Examples

```
getTaxonomy("Zamiaceae")
getTaxonomy(genus="Ceratozamia", depth.out=8)
getTaxonomy("Zamiaceae", "Ceratozamia", taxaType = 2)
```

---

meanPositiveValues	<i>Convert data into presence/absence data.</i>
--------------------	---

---

**Description**

Convert data into presence/absence data.

**Usage**

```
meanPositiveValues(x)
```

**Arguments**

x                      A vector of values.

**Value**

The average of all the positive values.

**Examples**

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

---

normalise	<i>Convert data into presence/absence data.</i>
-----------	---

---

**Description**

Convert data into presence/absence data.

**Usage**

```
normalise(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)
```

# Index

## \* datasets

- crest\_ex, [11](#)
- crest\_ex\_pse, [12](#)
- crest\_ex\_selection, [12](#)

- accClimateVariables, [2](#), [17](#), [18](#)
- accContinentNames, [2](#), [17](#), [18](#)
- accRealmNames, [3](#), [17](#), [18](#)

- calib\_clim\_space, [3](#), [15](#), [16](#)
- close\_db\_connection, [4](#)
- connect\_online, [4](#)
- convert2percentages, [5](#)
- convert2presenceAbsence, [6](#)
- crest, [6](#)
- crest.calibrate, [8](#)
- crest.get\_modern\_data, [9](#)
- crest.reconstruct, [10](#)
- crest\_ex, [11](#)
- crest\_ex\_pse, [12](#)
- crest\_ex\_selection, [12](#)
- crestObj, [13](#)

- dbRequest, [14](#)

- fit\_pdfsp, [3](#), [15](#)
- fit\_xrange, [16](#)

- getClimateSpace, [16](#)
- getDistribTaxa, [18](#)
- getTaxonID, [18](#), [19](#)
- getTaxonomy, [20](#)

- meanPositiveValues, [21](#)

- normalise, [21](#)