Package 'CoordinateCleaner'

January 22, 2019

Type Package

Title Automated Cleaning of Occurrence Records from Biological Collections

Version 2.0-7

Description Automated flagging of common spatial and temporal errors in biological and paleontological collection data, for the use in conservation, ecology and paleontology. Includes automated tests to easily flag (and exclude) records assigned to country or province centroid, the open ocean, the headquarters of the Global Biodiversity Information Facility, urban areas or the location of biodiversity institutions (museums, zoos, botanical gardens, universities). Furthermore identifies per species outlier coordinates, zero coordinates, identical latitude/longitude and invalid coordinates. Also implements an algorithm to identify data sets with a significant proportion of rounded coordinates. Especially suited for large data sets.

Language en-gb

License GPL-3

URL https://ropensci.github.io/CoordinateCleaner/

BugReports https://github.com/ropensci/CoordinateCleaner/issues

Depends R (>= 3.5.0)

Imports dplyr, geosphere, ggplot2, graphics, methods, raster, rgeos, rgdal, rnaturalearth, stats, sp, tidyselect, utils

LazyData true

RoxygenNote 6.1.1

Suggests countrycode, covr, knitr, maps, paleobioDB, rgbif, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Alexander Zizka [aut, cre],
Daniele Silvestro [ctb],
Tobias Andermann [ctb],
Josue Azevedo [ctb],
Camila Duarte Ritter [ctb],

Daniel Edler [ctb],
Harith Farooq [ctb],
Andrei Herdean [ctb],
Maria Ariza [ctb],
Ruud Scharn [ctb],
Sten Svanteson [ctb],
Niklas Wengstrom [ctb],
Vera Zizka [ctb],
Alexandre Antonelli [ctb],
Irene Steves [rev] (Irene reviewed the package for ropensci, see <https://github.com/ropensci/onboarding/issues/210>),
Francisco Rodriguez-Sanchez [rev] (Francisco reviewed the package for ropensci, see <https://github.com/ropensci/onboarding/issues/210>)

Maintainer Alexander Zizka <alexander.zizka@idiv.de>

Repository CRAN

Date/Publication 2019-01-22 15:50:20 UTC

R topics documented:

`CoordinateCleaner-package` *CoordinateCleaner*

Description

Automated Cleaning of Occurrence Records from Biological Collections

Details

Automated flagging of common spatial and temporal errors in biological and paleontological collection data, for the use in conservation, ecology and paleontology. Includes automated tests to easily flag (and exclude) records assigned to country or province centroid, the open ocean, the headquarters of the Global Biodiversity Information Facility, urban areas or the location of biodiversity institutions (museums, zoos, botanical gardens, universities). Furthermore identifies per species outlier coordinates, zero coordinates, identical latitude/longitude and invalid coordinates. Also implements an algorithm to identify data sets with a significant proportion of rounded coordinates. Especially suited for large data sets. See <https://ropensci.github.io/CoordinateCleaner/> for more details and tutorials.

Author(s)

Alexander Zizka, Daniele Silvestro, Tobias Andermann, Josue Azevedo, Camila Duarte Ritter, Daniel Edler, Harith Farooq, Andrei Herdean, Maria Ariza, Ruud Scharn, Sten Svantesson, Niklas Wengstrom, Vera Zizka

---

`buffland` *Global Coastlines buffered by 1 degree*

Description

A `SpatialPolygonsDataFrame` with global coastlines, with a 1 degree buffer to extent coastlines as alternative reference for `cc_sea`. Can be useful to identify species in the sea, without flagging records in mangroves, marshes, etc.

Source

http://www.naturalearthdata.com/downloads/10m-physical-vectors/

Examples

```
data("buffland")
```

cc_cap *Identify Coordinates in Vicinity of Country Capitals.*

Description

Removes or flags records within a certain radius around country capitals. Poorly geo-referenced occurrence records in biological databases are often erroneously geo-referenced to capitals.

Usage

```
cc_cap(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", buffer = 10000, geod = TRUE, ref = NULL,
  verify = FALSE, value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	character string. The column with the species identity. Only required if verify = TRUE.
buffer	The buffer around each capital coordinate (the centre of the city), where records should be flagged as problematic. Units depend on geod. Default = 10 kilometres.
geod	logical. If TRUE the radius around each capital is calculated based on a sphere, buffer is in meters and independent of latitude. If FALSE the radius is calculated assuming planar coordinates and varies slightly with latitude, in this case buffer is in degrees. Default = TRUE. See https://seethedatablog.wordpress.com/ for detail and credits.
ref	SpatialPointsDataFrame. Providing the geographic gazetteer. Can be any SpatialPointsDataFrame, but the structure must be identical to countryref. Default = countryref.
verify	logical. If TRUE records are only flagged if they are the only record in a given species flagged close to a given reference. If FALSE, the distance is the only criterion
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

cc_cap(x)
cc_cap(x, value = "flagged")
```

---

cc_cen *Identify Coordinates in Vicinity of Country and Province Centroids*

Description

Removes or flags records within a radius around the geographic centroids of political countries and provinces. Poorly geo-referenced occurrence records in biological databases are often erroneously geo-referenced to centroids.

Usage

```
cc_cen(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", buffer = 1000, geod = TRUE, test = "both",
  ref = NULL, verify = FALSE, value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	character string. The column with the species identity. Only required if verify = TRUE.
buffer	numerical. The buffer around each province or country centroid, where records should be flagged as problematic. Units depend on geod. Default = 1 kilometre.

geod	logical. If TRUE the radius around each capital is calculated based on a sphere, buffer is in meters and independent of latitude. If FALSE the radius is calculated assuming planar coordinates and varies slightly with latitude, in this case buffer is in degrees. Default = TRUE. See https://seethedatablog.wordpress.com/ for detail and credits.
test	a character string. Specifying the details of the test. One of c("both", "country", "provinces"). If both tests for country and province centroids.
ref	SpatialPointsDataFrame. Providing the geographic gazetteer. Can be any SpatialPointsDataFrame, but the structure must be identical to countryref. Default = countryref.
verify	logical. If TRUE records are only flagged if they are the only record in a given species flagged close to a given reference. If FALSE, the distance is the only criterion
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

cc_cen(x, geod = FALSE)

## Not run:
#' cc_inst(x, value = "flagged", buffer = 50000) #geod = T

## End(Not run)
```

cc_coun *Identify Coordinates Outside their Reported Country*

Description

Removes or flags mismatches between geographic coordinates and additional country information (usually this information is reliably reported with specimens). Such a mismatch can occur for example, if latitude and longitude are switched.

Usage

```
cc_coun(x, lon = "decimallongitude", lat = "decimallatitude",
  iso3 = "countrycode", value = "clean", ref = NULL,
  verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
iso3	a character string. The column with the country assignment of each record in three letter ISO code. Default = "countrycode".
value	character string. Defining the output value. See value.
ref	a SpatialPolygonsDataFrame. Providing the geographic gazetteer. Can be any SpatialPolygonsDataFrame, but the structure must be identical to `rnaturalearth::ne_countries(scal` Default = `rnaturalearth::ne_countries(scale = "medium")`
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

With the default reference, records are flagged if they fall outside the terrestrial territory of countries, hence records in territorial waters might be flagged. See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
## Not run:
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -20, 30),
                decimallatitude = runif(100, 35,60),
                countrycode = "RUS")

cc_coun(x, value = "flagged")#non-terrestrial records are flagged as wrong.

## End(Not run)
```

---

cc_dupl *Identify Duplicated Records*

Description

Removes or flags duplicated records based on species name and coordinates, as well as user-defined additional columns. True (specimen) duplicates or duplicates from the same species can make up the bulk of records in a biological collection database, but are undesirable for many analyses. Both can be flagged with this function, the former given enough additional information.

Usage

```
cc_dupl(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", additions = NULL, value = "clean",
  verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	a character string. The column with the species name. Default = "species".
additions	a vector of character strings. Additional columns to be included in the test for duplication. For example as below, collector name and collector number.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = sample(x = 0:10, size = 100, replace = TRUE),
                decimallatitude = sample(x = 0:10, size = 100, replace = TRUE),
                collector = "Bonpl",
                collector.number = c(1001, 354),
                collection = rep(c("K", "WAG","FR", "P", "S"), 20))

cc_dupl(x, value = "flagged")
cc_dupl(x, additions = c("collector", "collector.number"))
```

---

cc_equ *Identify Records with Identical lat/lon*

Description

Removes or flags records with equal latitude and longitude coordinates, either exact or absolute. Equal coordinates can often indicate data entry errors.

Usage

```
cc_equ(x, lon = "decimallongitude", lat = "decimallatitude",
  test = "absolute", value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
test	character string. Defines if coordinates are compared exactly ("identical") or on the absolute scale (i.e. -1 = 1, "absolute"). Default is to "absolute".
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

cc_equ(x)
cc_equ(x, value = "flagged")
```

---

cc_gbif *Identify Records Assigned to GBIF Headquarters*

Description

Removes or flags records within 0.5 degree radius around the GBIF headquarters in Copenhagen, DK.

Usage

```
cc_gbif(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", buffer = 1000, geod = TRUE, verify = FALSE,
  value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".

species	character string. The column with the species identity. Only required if verify = TRUE.
buffer	numerical. The buffer around the GBIF headquarters, where records should be flagged as problematic. Units depend on geod. Default = 100 m.
geod	logical. If TRUE the radius is calculated based on a sphere, buffer is in meters. If FALSE the radius is calculated in degrees. Default = T.
verify	logical. If TRUE records are only flagged if they are the only record in a given species flagged close to a given reference. If FALSE, the distance is the only criterion
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

Not recommended if working with records from Denmark or the Copenhagen area.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = "A",
                decimallongitude = c(12.58, 12.58),
                decimallatitude = c(55.67, 30.00))

cc_gbif(x)
cc_gbif(x, value = "flagged")
```

cc_inst *Identify Records in the Vicinity of Biodiversity Institutions*

Description

Removes or flags records assigned to the location of zoos, botanical gardens, herbaria, universities and museums, based on a global database of ~10,000 such biodiversity institutions. Coordinates from these locations can be related to data-entry errors, false automated geo-reference or individuals in captivity/horticulture.

Usage

```
cc_inst(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", buffer = 100, geod = TRUE, ref = NULL,
  verify = FALSE, verify_mltpl = 10, value = "clean",
  verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	character string. The column with the species identity. Only required if verify = TRUE.
buffer	numerical. The buffer around each institution, where records should be flagged as problematic, in decimal degrees. Default = 100m.
geod	logical. If TRUE the radius around each capital is calculated based on a sphere, buffer is in meters and independent of latitude. If FALSE the radius is calculated assuming planar coordinates and varies slightly with latitude, in this case buffer is in degrees. Default = TRUE. See https://seethedatablog.wordpress.com/ for detail and credits.
ref	SpatialPointsDataFrame. Providing the geographic gazetteer. Can be any SpatialPointsDataFrame, but the structure must be identical to institutions. Default = institutions
verify	logical. If TRUE, records close to institutions are only flagged, if there are no other records of the same species in the greater vicinity (a radius of buffer * verify_mltpl).
verify_mltpl	numerical. indicates the factor by which the radius for verify exceeds the radius of the initial test. Default = 10, which might be suitable if geod is TRUE, but might be too large otherwise.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

Note: the buffer radius is in degrees, thus will differ slightly between different latitudes.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

#large buffer for demonstration, using geod = FALSE for shorter runtime
cc_inst(x, value = "flagged", buffer = 10, geod = FALSE)

## Not run:
#' cc_inst(x, value = "flagged", buffer = 50000) #geod = T

## End(Not run)
```

---

cc_iucn *Identify Records Outside Natural Ranges*

Description

Removes or flags records outside of the provided natural range polygon, on a per species basis. Expects one entry per species. See the example or https://www.iucnredlist.org/resources/spatial-data-download for the required polygon structure.

Usage

```
cc_iucn(x, range, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", buffer = 0, value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
range	a SpatialPolygonsDataFrame of natural ranges for species in x. Must contain a column named as indicated by species. See details.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	a character string. The column with the species name. Default = "species".
buffer	numerical. The buffer around each species' range, from where records should be flagged as problematic, in decimal degrees. Default = 0.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

Download natural range maps in suitable format for amphibians, birds, mammals and reptiles from https://www.iucnredlist.org/resources/spatial-data-download. Note: the buffer radius is in degrees, thus will differ slightly between different latitudes.

Value

Depending on the 'value' argument, either a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_outl, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
require(sp)

x <- data.frame(species = c("A", "B"),
decimallongitude = runif(100, -170, 170),
decimallatitude = runif(100, -80,80))

range_species_A <- Polygon(cbind(c(-45,-45,-60,-60,-45),c(-10,-25,-25,-10,-10)))
range_species_B <- Polygon(cbind(c(15,15,32,32,15),c(10,-10,-10,10,10)))
range_A <- Polygons(list(range_species_A), ID = c("A"))
range_B <- Polygons(list(range_species_B), ID = c("B"))
range <- SpatialPolygons(list(range_A, range_B))
```

```
df <- data.frame(species = c("A", "B"), row.names = c("A", "B"))
range <- SpatialPolygonsDataFrame(range, data = as.data.frame(df))

cc_iucn(x = x, range = range, buffer = 10)
```

---

cc_outl *Identify Geographic Outliers in Species Distributions*

Description

Removes out or flags records that are outliers in geographic space according to the method defined via the method argument. Geographic outliers often represent erroneous coordinates, for example due to data entry errors, imprecise geo-references, individuals in horticulture/captivity.

Usage

```
cc_outl(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", method = "quantile", mltpl = 5, tdi = 1000,
  value = "clean", sampling_thresh = 0, verbose = TRUE,
  min_occs = 7)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	character string. The column with the species name. Default = "species".
method	character string. Defining the method for outlier selection. See details. One of "distance", "quantile", "mad". Default = "quantile".
mltpl	numeric. The multiplier of the interquartile range (method == 'quantile') or median absolute deviation (method == 'mad')to identify outliers. See details. Default = 5.
tdi	numeric. The minimum absolute distance (method == 'distance') of a record to all other records of a species to be identified as outlier, in km. See details. Default = 1000.
value	character string. Defining the output value. See value.
sampling_thresh	numeric. Cut off threshold for the sampling correction. Indicates the quantile of sampling in which outliers should be ignored. For instance, if sampling_thresh == 0.25, records in the 25 not be flagged as outliers. Default = 0 (no sampling correction).
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

`min_occs` Minimum number of geographically unique datapoints needed for a species to be tested. This is necessary for reliable outlier estimation. Species wit less than min_occs records will not be tested and the output value will be 'TRUE'. Default is to 7. If `method == 'distance'`, consider a lower threshold.

Details

The method for outlier identification depends on the `method` argument. If "outlier": a boxplot method is used and records are flagged as outliers if their *mean* distance to all other records of the same species is larger than mltpl * the interquartile range of the mean distance of all records of this species. If "mad": the median absolute deviation is used. In this case a record is flagged as outlier, if the *mean* distance to all other records of the same species is larger than the median of the mean distance of all points plus/minus the mad of the mean distances of all records of the species * mltpl. If "distance": records are flagged as outliers, if the *minimum* distance to the next record of the species is > `tdi`. For species with records from > 10000 unique locations a random sample of 1000 records is used for the distance matrix calculation. The test skipps species with less than `min_occs`, geographically unique records.

The likelihood of occurrence records being erroneous outliers is linked to the sampling effort in any given location. To account for this, the sampling_cor option fetches the number of occurrence records available from www.gbif.org, per country as a proxy of sampling effort. The outlier test (the mean distance) for each records is than weighted by the log transformed number of records per square kilometre in this country. See for https://ropensci.github.io/CoordinateCleaner/articles/Tutorial_geographic_outliers.html an example and further explanation of the outlier test.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_sea, cc_urb, cc_val, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

cc_outl(x)
cc_outl(x, method = "quantile", value = "flagged")
cc_outl(x, method = "distance", value = "flagged", tdi = 10000)
```

```
cc_outl(x, method = "distance", value = "flagged", tdi = 1000)
```

---

cc_sea *Identify Non-terrestrial Coordinates*

Description

Removes or flags coordinates outside the reference landmass. Can be used to restrict datasets to terrestrial taxa, or exclude records from the open ocean, when depending on the reference (see details). Often records of terrestrial taxa can be found in the open ocean, mostly due to switched latitude and longitude.

Usage

```
cc_sea(x, lon = "decimallongitude", lat = "decimallatitude",
  ref = NULL, scale = 110, value = "clean", speedup = TRUE,
  verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
ref	a SpatialPolygonsDataFrame. Providing the geographic gazetteer. Can be any SpatialPolygonsDataFrame, but the structure must be identical to rnaturalearth::ne_download(scale = 110, type = 'land', category = 'physical'). Default = rnaturalearth::ne_download(scale = 110, type = 'land', category = 'physical')
scale	the scale of the default reference, as downloaded from natural earth. Must be one of 10, 50, 110. Higher numbers equal higher detail. Default = 110.
value	character string. Defining the output value. See value.
speedup	logical. Using heuristic to speed up the analysis for large data sets with many records per location.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

In some cases flagging records close of the coastline is not recommendable, because of the low precision of the reference dataset, minor GPS imprecision or because a dataset might include coast or marshland species. If you only want to flag records in the open ocean, consider using a buffered landmass reference, e.g.: buffland.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: `cc_cap`, `cc_cen`, `cc_coun`, `cc_dupl`, `cc_equ`, `cc_gbif`, `cc_inst`, `cc_iucn`, `cc_outl`, `cc_urb`, `cc_val`, `cc_zero`

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(10, -30, 30),
                decimallatitude = runif(10, -30, 30))

cc_sea(x, value = "flagged")
```

---

cc_urb *Identify Records Inside Urban Areas*

Description

Removes or flags records from inside urban areas, based on a geographic gazetteer. Often records from large databases span substantial time periods (centuries) and old records might represent habitats which today are replaced by city area.

Usage

```
cc_urb(x, lon = "decimallongitude", lat = "decimallatitude",
  ref = NULL, value = "clean", verbose = TRUE)
```

Arguments

`x`	data.frame. Containing geographical coordinates and species names.
`lon`	character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	character string. The column with the latitude coordinates. Default = "decimallatitude".

ref	a SpatialPolygonsDataFrame. Providing the geographic gazetteer with the urban areas. See details. By default rnaturalearth::ne_download(scale = 'medium', type = 'urban_areas'). Can be any SpatialPolygonsDataframe, but the structure must be identical to rnaturalearth::ne_download().
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_val, cc_zero

Examples

```
## Not run:
x <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90))

cc_urb(x)
cc_urb(x, value = "flagged")

## End(Not run)
```

---

cc_val *Identify Invalid lat/lon Coordinates*

Description

Removes or flags non-numeric and not available coordinates as well as lat >90, la <-90, lon > 180 and lon < -180 are flagged.

Usage

```
cc_val(x, lon = "decimallongitude", lat = "decimallatitude",
  value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

This test is obligatory before running any further tests of CoordinateCleaner, as additional tests only run with valid coordinates.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_zero

Examples

```
x <- data.frame(species = letters[1:10],
                decimallongitude = c(runif(106, -180, 180), NA, "13W33'", "67,09", 305),
                decimallatitude = runif(110, -90,90))

cc_val(x)
cc_val(x, value = "flagged")
```

cc_zero *Identify Zero Coordinates*

Description

Removes or flags records with either zero longitude or latitude and a radius around the point at zero longitude and zero latitude. These problems are often due to erroneous data-entry or geo-referencing and can lead to typical patterns of high diversity around the equator.

Usage

```
cc_zero(x, lon = "decimallongitude", lat = "decimallatitude",
  buffer = 0.5, value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
buffer	numerical. The buffer around the 0/0 point, where records should be flagged as problematic, in decimal degrees. Default = 0.1.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Coordinates: cc_cap, cc_cen, cc_coun, cc_dupl, cc_equ, cc_gbif, cc_inst, cc_iucn, cc_outl, cc_sea, cc_urb, cc_val

Examples

```
x <- data.frame(species = "A",
                decimallongitude = c(0,34.84, 0, 33.98),
                decimallatitude = c(23.08, 0, 0, 15.98))

cc_zero(x)
cc_zero(x, value = "flagged")
```

---

cd_ddmm *Identify Datasets with a Degree Conversion Error*

Description

This test flags datasets where a significant fraction of records has been subject to a common degree minute to decimal degree conversion error, where the degree sign is recognized as decimal delimiter.

Usage

```
cd_ddmm(x, lon = "decimallongitude", lat = "decimallatitude",
  ds = "dataset", pvalue = 0.025, diff = 1, mat_size = 1000,
  min_span = 2, value = "clean", verbose = TRUE,
  diagnostic = FALSE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
ds	a character string. The column with the dataset of each record. In case x should be treated as a single dataset, identical for all records. Default = "dataset".
pvalue	numeric. The p-value for the one-sided t-test to flag the test as passed or not. Both ddmm.pvalue and diff must be met. Default = 0.025.
diff	numeric. The threshold difference for the ddmm test. Indicates by which fraction the records with decimals below 0.6 must outnumber the records with decimals above 0.6. Default = 1
mat_size	numeric. The size of the matrix for the binomial test. Must be changed in decimals (e.g. 100, 1000, 10000). Adapt to dataset size, generally 100 is better for datasets < 10000 records, 1000 is better for datasets with 10000 - 1M records. Higher values also work reasonably well for smaller datasets, therefore, default = 1000. For large datasets try 10000.
min_span	numeric. The minimum geographic extent of datasets to be tested. Default = 2.

value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.
diagnostic	logical. If TRUE plots the analyses matrix for each dataset.

Details

If the degree sign is recognized as decimal delimiter during coordinate conversion, no coordinate decimals above 0.59 (59') are possible. The test here uses a binomial test to test if a significant proportion of records in a dataset have been subject to this problem. The test is best adjusted via the diff argument. The lower diff, the stricter the test. Also scales with dataset size. Empirically, for datasets with < 5,000 unique coordinate records `diff = 0.1` has proven reasonable flagging most datasets with >25% problematic records and all dataset with >50% problematic records. For datasets between 5,000 and 100,000 geographic unique records `diff = 0.01` is recommended, for datasets between 100,000 and 1 M records diff = 0.001, and so on.

Value

Depending on the 'value' argument, either a `data.frame` with summary statistics and flags for each dataset ("dataset") or a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flags"), with TRUE = test passed and FALSE = test failed/potentially problematic. Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Datasets: cd_round

Examples

```
clean <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90),
                dataset = "FR")

cd_ddmm(x = clean, value = "flagged")

#problematic dataset
lon <- sample(0:180, size = 100, replace = TRUE) + runif(100, 0,0.59)
lat <- sample(0:90, size = 100, replace = TRUE) + runif(100, 0,0.59)

prob <-  data.frame(species = letters[1:10],
                decimallongitude = lon,
                decimallatitude = lat,
                dataset = "FR")

cd_ddmm(x = prob, value = "flagged")
```

cd_round *Identify Datasets with Rasterized Coordinates*

Description

Flags datasets with periodicity patterns indicative of a rasterized (lattice) collection scheme, as often obtain from e.g. atlas data. Using a combination of autocorrelation and sliding-window outlier detection to identify periodicity patterns in the data. See https://ropensci.github.io/CoordinateCleaner/articles/Background_dataset_level_cleaning.html for further details and a description of the algorithm

Usage

```
cd_round(x, lon = "decimallongitude", lat = "decimallatitude",
  ds = "dataset", T1 = 7, reg_out_thresh = 2, reg_dist_min = 0.1,
  reg_dist_max = 2, min_unique_ds_size = 4, graphs = TRUE,
  test = "both", value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
ds	a character string. The column with the dataset of each record. In case x should be treated as a single dataset, identical for all records. Default = "dataset".
T1	numeric. The threshold for outlier detection in a in an interquantile range based test. This is the major parameter to specify the sensitivity of the test: lower values, equal higher detection rate. Values between 7-11 are recommended. Default = 7.
reg_out_thresh	numeric. Threshold on the number of equal distances between outlier points. See details. Default = 2.
reg_dist_min	numeric. The minimum detection distance between outliers in degrees (the minimum resolution of grids that will be flagged). Default = 0.1.
reg_dist_max	numeric. The maximum detection distance between outliers in degrees (the maximum resolution of grids that will be flagged). Default = 2.
min_unique_ds_size	numeric. The minimum number of unique locations (values in the tested column) for datasets to be included in the test. Default = 4.
graphs	logical. If TRUE, diagnostic plots are produced. Default = TRUE.
test	character string. Indicates which column to test. Either "lat" for latitude, "lon" for longitude, or "both" for both. In the latter case datasets are only flagged if both test are failed. Default = "both"

value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a data.frame with summary statistics and flags for each dataset ("dataset") or a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic. Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Datasets: cd_ddmm

Examples

```
#simulate bias grid, one degree resolution, 10% error on a 1000 records dataset
#simulate biased fraction of the data, grid resolution = 1 degree
#simulate non-biased fraction of the data
  bi <- sample(3 + 0:5, size = 100, replace = TRUE)
  mu <- runif(3, 0, 15)
  sig <- runif(3, 0.1, 5)
  cl <- rnorm(n = 900, mean = mu, sd = sig)
  lon <- c(cl, bi)

  bi <- sample(9:13, size = 100, replace = TRUE)
  mu <- runif(3, 0, 15)
  sig <- runif(3, 0.1, 5)
  cl <- rnorm(n = 900, mean = mu, sd = sig)
  lat <- c(cl, bi)

  #add biased data

  inp <- data.frame(decimallongitude = lon,
                    decimallatitude = lat,
                    dataset = "test")


  #run test
  ## Not run:
  cd_round(inp, value = "dataset")

## End(Not run)
```

cf_age *Identify Fossils with Outlier Age*

Description

Removes or flags records that are temporal outliers based on interquantile ranges.

Usage

```
cf_age(x, lon = "decimallongitude", lat = "decimallatitude",
  min_age = "min_ma", max_age = "max_ma", taxon = "accepted_name",
  method = "quantile", size_thresh = 7, mltpl = 5, replicates = 5,
  flag_thresh = 0.5, uniq_loc = FALSE, value = "clean",
  verbose = TRUE)
```

Arguments

x	data.frame. Containing fossil records with taxon names, ages, and geographic coordinates.
lon	character string. The column with the longitude coordinates. To identify unique records if uniq_loc = TRUE. Default = "decimallongitude".
lat	character string. The column with the longitude coordinates. Default = "decimallatitude". To identify unique records if uniq_loc = T.
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
taxon	character string. The column with the taxon name. If "", searches for outliers over the entire dataset, otherwise per specified taxon. Default = "accepted_name".
method	character string. Defining the method for outlier selection. See details. Either "quantile" or "mad". Default = "quantile".
size_thresh	numeric. The minimum number of records needed for a dataset to be tested. Default = 10.
mltpl	numeric. The multiplier of the interquartile range (method == 'quantile') or median absolute deviation (method == 'mad') to identify outliers. See details. Default = 5.
replicates	numeric. The number of replications for the distance matrix calculation. See details. Default = 5.
flag_thresh	numeric. The fraction of passed replicates necessary to pass the test. See details. Default = 0.5.
uniq_loc	logical. If TRUE only single records per location and time point (and taxon if taxon != "") are used for the outlier testing. Default = T.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

The outlier detection is based on an interquantile range test. A temporal distance matrix among all records is calculated based on a single point selected by random between the minimum and maximum age for each record. The mean distance for each point to all neighbours is calculated and the sum of these distances is then tested against the interquantile range and flagged as an outlier if $x > IQR(x) + q_75 * mltpl$. The test is replicated 'replicates' times, to account for dating uncertainty. Records are flagged as outliers if they are flagged by a fraction of more than 'flag.thresh' replicates. Only datasets/taxa comprising more than 'size_thresh' records are tested. Distance are calculated as Euclidean distance.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other fossils: `cf_equal`, `cf_outl`, `cf_range`

Examples

```
minages <- c(runif(n = 11, min = 10, max = 25), 62.5)
x <- data.frame(species = c(letters[1:10], rep("z", 2)),
                min_ma = minages,
                max_ma = c(minages[1:11] + runif(n = 11, min = 0, max = 5), 65))

cf_age(x, value = "flagged", taxon = "")

# unique locations only
x <- data.frame(species = c(letters[1:10], rep("z", 2)),
                decimallongitude = c(runif(n = 10, min = 4, max = 16), 75, 7),
                decimallatitude = c(runif(n = 12, min = -5, max = 5)),
                min_ma = minages,
                max_ma = c(minages[1:11] + runif(n = 11, min = 0, max = 5), 65))

cf_age(x, value = "flagged", taxon = "", uniq_loc = TRUE)
```

cf_equal *Identify Fossils with equal min and max age*

Description

Removes or flags records with equal minimum and maximum age.

Usage

```
cf_equal(x, min_age = "min_ma", max_age = "max_ma", value = "clean",
  verbose = TRUE)
```

Arguments

x	data.frame. Containing fossil records with taxon names, ages, and geographic coordinates.
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other fossils: cf_age, cf_outl, cf_range

Examples

```
minages <- runif(n = 10, min = 0.1, max = 25)
x <- data.frame(species = letters[1:10],
                min_ma = minages,
                max_ma = minages + runif(n = 10, min = 0, max = 10))
x <- rbind(x, data.frame(species = "z",
                min_ma = 5,
                max_ma = 5))

cf_equal(x, value = "flagged")
```

cf_outl *Identify Outlier Records in Space and Time*

Description

Removes or flags records of fossils that are spatio-temporal outliers based on interquantile ranges. Records are flagged if they are either extreme in time or space, or both.

Usage

```
cf_outl(x, lon = "decimallongitude", lat = "decimallatitude",
  min_age = "min_ma", max_age = "max_ma", taxon = "accepted_name",
  method = "quantile", size_thresh = 7, mltpl = 5, replicates = 5,
  flag_thresh = 0.5, uniq_loc = FALSE, value = "clean",
  verbose = TRUE)
```

Arguments

x	data.frame. Containing fossil records with taxon names, ages, and geographic coordinates.
lon	character string. The column with the longitude coordinates. To identify unique records if uniq_loc = TRUE. Default = "decimallongitude".
lat	character string. The column with the longitude coordinates. Default = "decimallatitude". To identify unique records if uniq_loc = T.
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
taxon	character string. The column with the taxon name. If "", searches for outliers over the entire dataset, otherwise per specified taxon. Default = "accepted_name".
method	character string. Defining the method for outlier selection. See details. Either "quantile" or "mad". Default = "quantile".
size_thresh	numeric. The minimum number of records needed for a dataset to be tested. Default = 10.
mltpl	numeric. The multiplier of the interquartile range (method == 'quantile') or median absolute deviation (method == 'mad') to identify outliers. See details. Default = 5.
replicates	numeric. The number of replications for the distance matrix calculation. See details. Default = 5.
flag_thresh	numeric. The fraction of passed replicates necessary to pass the test. See details. Default = 0.5.
uniq_loc	logical. If TRUE only single records per location and time point (and taxon if taxon != "") are used for the outlier testing. Default = T.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

The outlier detection is based on an interquantile range test. In a first step a distance matrix of geographic distances among all records is calculate. Subsequently a similar distance matrix of temporal distances among all records is calculated based on a single point selected by random between the minimum and maximum age for each record. The mean distance for each point to all neighbours is calculated for both matrices and spatial and temporal distances are scaled to the same range. The sum of these distanced is then tested against the interquantile range and flagged as an outlier if $x > IQR(x) + q_75 * mltpl$. The test is replicated 'replicates' times, to account for temporal uncertainty. Records are flagged as outliers if they are flagged by a fraction of more than 'flag.thres' replicates. Only datasets/taxa comprising more than 'size_thresh' records are tested. Note that geographic distances are calculated as geospheric distances for datasets (or taxa) with less than 10,000 records and approximated as Euclidean distances for datasets/taxa with 10,000 to 25,000 records. Datasets/taxa comprising more than 25,000 records are skipped.

Value

Depending on the 'value' argument, either a `data.frame` containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other fossils: `cf_age`, `cf_equal`, `cf_range`

Examples

```
minages <- c(runif(n = 11, min = 10, max = 25), 62.5)
x <- data.frame(species = c(letters[1:10], rep("z", 2)),
                lng = c(runif(n = 10, min = 4, max = 16), 75, 7),
                lat = c(runif(n = 12, min = -5, max = 5)),
                min_ma = minages,
                max_ma = c(minages[1:11] + runif(n = 11, min = 0, max = 5), 65))

cf_outl(x, value = "flagged", taxon = "")
```

`cf_range` *Identify Fossils with Extreme Age Ranges*

Description

Removes or flags records with an unexpectedly large temporal range, based on a quantile outlier test.

Usage

```
cf_range(x, lon = "decimallongitude", lat = "decimallatitude",
  min_age = "min_ma", max_age = "max_ma", taxon = "accepted_name",
  method = "quantile", mltpl = 5, size_thresh = 7, max_range = 500,
  uniq_loc = FALSE, value = "clean", verbose = TRUE)
```

Arguments

x	data.frame. Containing fossil records with taxon names, ages, and geographic coordinates.
lon	character string. The column with the longitude coordinates. To identify unique records if uniq_loc = TRUE. Default = "decimallongitude".
lat	character string. The column with the longitude coordinates. Default = "decimallatitude". To identify unique records if uniq_loc = T.
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
taxon	character string. The column with the taxon name. If "", searches for outliers over the entire dataset, otherwise per specified taxon. Default = "accepted_name".
method	character string. Defining the method for outlier selection. See details. Either "quantile" or "mad". Default = "quantile".
mltpl	numeric. The multiplier of the interquartile range (method == 'quantile') or median absolute deviation (method == 'mad') to identify outliers. See details. Default = 5.
size_thresh	numeric. The minimum number of records needed for a dataset to be tested. Default = 10.
max_range	numeric. A absolute maximum time interval between min age and max age. Only relevant for method = "time".
uniq_loc	logical. If TRUE only single records per location and time point (and taxon if taxon != "") are used for the outlier testing. Default = T.
value	character string. Defining the output value. See value.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Value

Depending on the 'value' argument, either a data.frame containing the records considered correct by the test ("clean") or a logical vector ("flagged"), with TRUE = test passed and FALSE = test failed/potentially problematic . Default = "clean".

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other fossils: cf_age, cf_equal, cf_outl

Examples

```
minages <- runif(n = 11, min = 0.1, max = 25)
x <- data.frame(species = c(letters[1:10], "z"),
                lng = c(runif(n = 9, min = 4, max = 16), 75, 7),
                lat = c(runif(n = 11, min = -5, max = 5)),
                min_ma = minages,
                max_ma = minages + c(runif(n = 10, min = 0, max = 5), 25))

cf_range(x, value = "flagged", taxon = "")
```

---

clean_coordinates *Geographic Cleaning of Coordinates from Biologic Collections*

Description

Cleaning geographic coordinates by multiple empirical tests to flag potentially erroneous coordinates, addressing issues common in biological collection databases.

Usage

```
clean_coordinates(x, lon = "decimallongitude", lat = "decimallatitude",
  species = "species", countries = NULL, tests = c("capitals",
  "centroids", "equal", "gbif", "institutions", "outliers", "seas",
  "zeros"), capitals_rad = 10000, centroids_rad = 1000,
  centroids_detail = "both", inst_rad = 100,
  outliers_method = "quantile", outliers_mtp = 5, outliers_td = 1000,
  outliers_size = 7, range_rad = 0, zeros_rad = 0.5,
  capitals_ref = NULL, centroids_ref = NULL, country_ref = NULL,
  inst_ref = NULL, range_ref = NULL, seas_ref = NULL,
  seas_scale = 50, urban_ref = NULL, value = "spatialvalid",
  verbose = TRUE, report = FALSE)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
species	a character string. A vector of the same length as rows in x, with the species identity for each record. If missing, the outliers test is skipped.
countries	a character string. The column with the country assignment of each record in three letter ISO code. Default = "countrycode". If missing, the countries test is skipped.

tests	a vector of character strings, indicating which tests to run. See details for all tests available. Default = c("capitals", "centroids", "equal", "gbif", "institutions", "outliers", "seas", "zeros")
capitals_rad	numeric. The radius around capital coordinates in meters. Default = 10000.
centroids_rad	numeric. The radius around capital coordinates in meters. Default = 1000.
centroids_detail	a character string. If set to 'country' only country (adm-0) centroids are tested, if set to 'provinces' only province (adm-1) centroids are tested. Default = 'both'.
inst_rad	numeric. The radius around biodiversity institutions coordinates in metres. Default = 100.
outliers_method	The method used for outlier testing. See details.
outliers_mtp	numeric. The multiplier for the interquartile range of the outlier test. If NULL outliers.td is used. Default = 5.
outliers_td	numeric. The minimum distance of a record to all other records of a species to be identified as outlier, in km. Default = 1000.
outliers_size	numerical. The minimum number of records in a dataset to run the taxon-specific outlier test. Default = 7.
range_rad	buffer around natural ranges. Default = 0.
zeros_rad	numeric. The radius around 0/0 in degrees. Default = 0.5.
capitals_ref	a data.frame with alternative reference data for the country capitals test. If missing, the countryref dataset is used. Alternatives must be identical in structure.
centroids_ref	a data.frame with alternative reference data for the centroid test. If NULL, the countryref dataset is used. Alternatives must be identical in structure.
country_ref	a SpatialPolygonsDataFrame as alternative reference for the countries test. If NULL, the rnaturalearth:ne_countries('medium') dataset is used.
inst_ref	a data.frame with alternative reference data for the biodiversity institution test. If NULL, the institutions dataset is used. Alternatives must be identical in structure.
range_ref	a SpatialPolygonsDataFrame of species natural ranges. Required to include the 'ranges' test. See cc_iucn for details.
seas_ref	a SpatialPolygonsDataFrame as alternative reference for the seas test. If NULL, the rnaturalearth::ne_download(scale = 50, type = 'land', category = 'physical') dataset is used.
seas_scale	The scale of the default landmass reference. Must be one of 10, 50, 110. Higher numbers equal higher detail. Default = 50.
urban_ref	a SpatialPolygonsDataFrame as alternative reference for the urban test. If NULL, the test is skipped. See details for a reference gazetteers.
value	a character string defining the output value. See the value section for details. one of 'spatialvalid', 'summary', 'cleaned'. Default = 'spatialvalid'.
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

`report` logical or character. If TRUE a report file is written to the working directory, summarizing the cleaning results. If a character, the path to which the file should be written. Default = FALSE.

Details

The function needs all coordinates to be formally valid according to WGS84. If the data contains invalid coordinates, the function will stop and return a vector flagging the invalid records. TRUE = non-problematic coordinate, FALSE = potentially problematic coordinates.

- capitals tests a radius around adm-0 capitals. The radius is `capitals_rad`.
- centroids tests a radius around country centroids. The radius is `centroids_rad`.
- countries tests if coordinates are from the country indicated in the country column. *Switched off by default.*
- duplicates tests for duplicate records. This checks for identical coordinates or if a species vector is provided for identical coordinates within a species. All but the first records are flagged as duplicates. *Switched off by default.*
- equal tests for equal absolute longitude and latitude.
- gbif tests a one-degree radius around the GBIF headquarters in Copenhagen, Denmark.
- institutions tests a radius around known biodiversity institutions from `instiutions`. The radius is `inst_rad`.
- outliers tests each species for outlier records. Depending on the `outliers_mtp` and `outliers.td` arguments either flags records that are a minimum distance away from all other records of this species (`outliers_td`) or records that are outside a multiple of the interquartile range of minimum distances to the next neighbour of this species (`outliers_mtp`). Three different methods are available for the outlier test: "If “outlier” a boxplot method is used and records are flagged as outliers if their *mean* distance to all other records of the same species is larger than mltpl * the interquartile range of the mean distance of all records of this species. If “mad” the median absolute deviation is used. In this case a record is flagged as outlier, if the *mean* distance to all other records of the same species is larger than the median of the mean distance of all points plus/minus the mad of the mean distances of all records of the species * mltpl. If “distance” records are flagged as outliers, if the *minimum* distance to the next record of the species is > `tdi`.
- ranges tests if records fall within provided natural range polygons on a per species basis. See `cc_iucn` for details.
- seas tests if coordinates fall into the ocean.
- urban tests if coordinates are from urban areas. *Switched off by default*
- validity checks if coordinates correspond to a lat/lon coordinate reference system. This test is always on, since all records need to pass for any other test to run.
- zeros tests for plain zeros, equal latitude and longitude and a radius around the point 0/0. The radius is `zeros.rad`.

Value

Depending on the output argument:

"spatialvalid" an object of class `spatialvalid` similar to x with one column added for each test. TRUE = clean coordinate entry, FALSE = potentially problematic coordinate entries. The .summary column is FALSE if any test flagged the respective coordinate.

"flagged" a logical vector with the same order as the input data summarizing the results of all test. TRUE = clean coordinate, FALSE = potentially problematic (= at least one test failed).

"clean" a `data.frame` similar to x with potentially problematic records removed

Note

Always tests for coordinate validity: non-numeric or missing coordinates and coordinates exceeding the global extent (lon/lat, WGS84). See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Wrapper functions: clean_dataset, clean_fossils

Examples

```
exmpl <- data.frame(species = sample(letters, size = 250, replace = TRUE),
                    decimallongitude = runif(250, min = 42, max = 51),
                    decimallatitude = runif(250, min = -26, max = -11))

test <- clean_coordinates(x = exmpl,
                          tests = c("equal"))

## Not run:
#run more tests
test <- clean_coordinates(x = exmpl,
                          tests = c("capitals",
                          "centroids","equal",
                          "gbif", "institutions",
                          "outliers", "seas",
                          "zeros"))

## End(Not run)


summary(test)
```

`clean_dataset` *Coordinate Cleaning using Dataset Properties*

Description

Tests for problems associated with coordinate conversions and rounding, based on dataset properties. Includes test to identify contributing datasets with potential errors with converting ddmm to dd.dd, and periodicity in the data decimals indicating rounding or a raster basis linked to low coordinate precision. Specifically:

- ddmm tests for erroneous conversion from a degree minute format (ddmm) to a decimal degree (dd.dd) format
- periodicity test for periodicity in the data, which can indicate imprecise coordinates, due to rounding or rasterization.

Usage

```
clean_dataset(x, lon = "decimallongitude", lat = "decimallatitude",
  ds = "dataset", tests = c("ddmm", "periodicity"),
  value = "dataset", verbose = TRUE, ...)
```

Arguments

x	data.frame. Containing geographical coordinates and species names.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
ds	a character string. The column with the dataset of each record. In case x should be treated as a single dataset, identical for all records. Default = "dataset".
tests	a vector of character strings, indicating which tests to run. See details for all tests available. Default = c("ddmm", "periodicity")
value	a character string. Defining the output value. See value. Default = "dataset".
verbose	logical. If TRUE reports the name of the test and the number of records flagged.
...	additional arguments to be passed to cd_ddmm and cd_round to customize test sensitivity.

Details

These tests are based on the statistical distribution of coordinates and their decimals within datasets of geographic distribution records to identify datasets with potential errors/biases. Three potential error sources can be identified. The ddmm flag tests for the particular pattern that emerges if geographical coordinates in a degree minute annotation are transferred into decimal degrees, simply replacing the degree symbol with the decimal point. This kind of problem has been observed by in older datasets first recorded on paper using typewriters, where e.g. a floating point was used as symbol for degrees. The function uses a binomial test to check if more records than expected have decimals below 0.6 (which is the maximum that can be obtained in minutes, as one degree has 60 minutes) and if the number of these records is higher than those above 0.59 by a certain proportion. The periodicity test uses rate estimation in a Poisson process to estimate if there is periodicity in the decimals of a dataset (as would be expected by for example rounding or data that was collected in a raster format) and if there is an over proportional number of records with the decimal 0

(full degrees) which indicates rounding and thus low precision. The default values are empirically optimized by with GBIF data, but should probably be adapted.

Value

Depending on the 'value' argument:

"dataset" a `data.frame` with the the test summary statistics for each dataset in `x`

"clean" a `data.frame` containing only records from datasets in `x` that passed the tests

"flagged" a logical vector of the same length as rows in `x`, with TRUE = test passed and FALSE = test failed/potentially problematic.

Note

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

cd_ddmm cd_round

Other Wrapper functions: clean_coordinates, clean_fossils

Examples

```
#Create test dataset
clean <- data.frame(dataset = rep("clean", 1000),
                    decimallongitude = runif(min = -43, max = -40, n = 1000),
                    decimallatitude = runif(min = -13, max = -10, n = 1000))

bias.long <- c(round(runif(min = -42, max = -40, n = 500), 1),
               round(runif(min = -42, max = -40, n = 300), 0),
               runif(min = -42, max = -40, n = 200))
bias.lat <- c(round(runif(min = -12, max = -10, n = 500), 1),
              round(runif(min = -12, max = -10, n = 300), 0),
              runif(min = -12, max = -10, n = 200))
bias <- data.frame(dataset = rep("biased", 1000),
                   decimallongitude = bias.long,
                   decimallatitude = bias.lat)
test <- rbind(clean, bias)

## Not run:
#run clean_dataset
flags <- clean_dataset(test)

#check problems
#clean
hist(test[test$dataset == rownames(flags[flags$summary,]), "decimallongitude"])
#biased
hist(test[test$dataset == rownames(flags[!flags$summary,]), "decimallongitude"])


## End(Not run)
```

clean_fossils *Geographic and Temporal Cleaning of Records from Fossil Collections*

Description

Cleaning records by multiple empirical tests to flag potentially erroneous coordinates and timespans, addressing issues common in fossil collection databases. Individual tests can be activated via the tests argument:

Usage

```
clean_fossils(x, lon = "lng", lat = "lat", min_age = "min_ma",
  max_age = "max_ma", taxon = "accepted_name", tests = c("agesequal",
  "centroids", "equal", "gbif", "institutions", "spatiotemp", "temprange",
  "validity", "zeros"), countries = NULL, centroids_rad = 0.05,
  centroids_detail = "both", inst_rad = 0.001,
  outliers_method = "quantile", outliers_threshold = 5,
  outliers_size = 7, outliers_replicates = 5, zeros_rad = 0.5,
  centroids_ref = NULL, country_ref = NULL, inst_ref = NULL,
  value = "spatialvalid", verbose = TRUE, report = FALSE)
```

Arguments

x	data.frame. Containing fossil records, containing taxon names, ages, and geographic coordinates..
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
taxon	character string. The column with the taxon name. If "", searches for outliers over the entire dataset, otherwise per specified taxon. Default = "accepted_name".
tests	vector of character strings, indicating which tests to run. See details for all tests available. Default = c("centroids", "equal", "gbif", "institutions", "temprange", "spatiotemp", "agesequal", "zeros")
countries	a character string. The column with the country assignment of each record in three letter ISO code. Default = "countrycode". If missing, the countries test is skipped.
centroids_rad	numeric. The radius around capital coordinates in meters. Default = 1000.
centroids_detail	a character string. If set to 'country' only country (adm-0) centroids are tested, if set to 'provinces' only province (adm-1) centroids are tested. Default = 'both'.

`inst_rad` numeric. The radius around biodiversity institutions coordinates in metres. Default = 100.

`outliers_method` The method used for outlier testing. See details.

`outliers_threshold` numerical. The multiplier for the interquantile range for outlier detection. The higher the number, the more conservative the outlier tests. See `cf_outl` for details. Default = 3.

`outliers_size` numerical. The minimum number of records in a dataset to run the taxon-specific outlier test. Default = 7.

`outliers_replicates` numeric. The number of replications for the distance matrix calculation. See details. Default = 5.

`zeros_rad` numeric. The radius around 0/0 in degrees. Default = 0.5.

`centroids_ref` a `data.frame` with alternative reference data for the centroid test. If NULL, the `countryref` dataset is used. Alternatives must be identical in structure.

`country_ref` a `SpatialPolygonsDataFrame` as alternative reference for the countries test. If NULL, the `rnaturalearth:ne_countries('medium')` dataset is used.

`inst_ref` a `data.frame` with alternative reference data for the biodiversity institution test. If NULL, the `institutions` dataset is used. Alternatives must be identical in structure.

`value` a character string defining the output value. See the value section for details. one of 'spatialvalid', 'summary', 'cleaned'. Default = '`spatialvalid`'.

`verbose` logical. If TRUE reports the name of the test and the number of records flagged.

`report` logical or character. If TRUE a report file is written to the working directory, summarizing the cleaning results. If a character, the path to which the file should be written. Default = FALSE.

Details

- agesequal tests for equal minimum and maximum age.
- centroids tests a radius around country centroids. The radius is `centroids_rad`.
- countries tests if coordinates are from the country indicated in the country column. *Switched off by default.*
- equal tests for equal absolute longitude and latitude.
- gbif tests a one-degree radius around the GBIF headquarters in Copenhagen, Denmark.
- institutions tests a radius around known biodiversity institutions from `instiutions`. The radius is `inst_rad`.
- spatiotemp test for records which are outlier in time and space. See below for details.
- temprange tests for records with unexpectedly large temporal ranges, using a quantile-based outlier test.
- validity checks if coordinates correspond to a lat/lon coordinate reference system. This test is always on, since all records need to pass for any other test to run.

- zeros tests for plain zeros, equal latitude and longitude and a radius around the point 0/0. The radius is `zeros_rad`. The outlier detection in 'spatiotemp' is based on an interquantile range test. In a first step a distance matrix of geographic distances among all records is calculate. Subsequently a similar distance matrix of temporal distances among all records is calculated based on a single point selected by random between the minimum and maximum age for each record. The mean distance for each point to all neighbors is calculated for both matrices and spatial and temporal distances are scaled to the same range. The sum of these distanced is then tested against the interquantile range and flagged as an outlier if $x > IQR(x) + q_7 5 * mltpl$. The test is replicated 'replicates' times, to account for temporal uncertainty. Records are flagged as outliers if they are flagged by a fraction of more than 'flag_thresh' replicates. Only datasets/taxa comprising more than 'size.thresh' records are tested. Note that geographic distances are calculated as geospheric distances for datasets (or taxa) with less than 10,000 records and approximated as Euclidean distances for datasets/taxa with 10,000 to 25,000 records. Datasets/taxa comprising more than 25,000 records are skipped.

Value

Depending on the output argument:

"spatialvalid" an object of class `spatialvalid` similar to x with one column added for each test. TRUE = clean coordinate entry, FALSE = potentially problematic coordinate entries. The .summary column is FALSE if any test flagged the respective coordinate.

"flagged" a logical vector with the same order as the input data summarizing the results of all test. TRUE = clean coordinate, FALSE = potentially problematic (= at least one test failed).

"clean" a `data.frame` similar to x with potentially problematic records removed

Note

Always tests for coordinate validity: non-numeric or missing coordinates and coordinates exceeding the global extent (lon/lat, WGS84).

See https://ropensci.github.io/CoordinateCleaner/ for more details and tutorials.

See Also

Other Wrapper functions: clean_coordinates, clean_dataset

Examples

```
minages <- runif(250, 0, 65)
exmpl <- data.frame(accepted_name = sample(letters, size = 250, replace = TRUE),
                    lng = runif(250, min = 42, max = 51),
                    lat = runif(250, min = -26, max = -11),
                    min_ma = minages,
                    max_ma = minages + runif(250, 0.1, 65))

test <- clean_fossils(x = exmpl)

summary(test)
```

CoordinateCleaner-deprecated *Deprecated functions in CoordinateCleaener*

Description

These functions still work but will be removed (defunct) in the next version.

Details

- `CleanCoordinates`: This function is deprecated, and will be removed in the next version of this package. Use `clean_coordinates` instead
- `CleanCoordinatesDS`: This function is deprecated, and will be removed in the next version of this package. Use `clean_dataset` instead
- `CleanCoordinatesFOS`: This function is deprecated, and will be removed in the next version of this package. Use `clean_fossils` instead

countryref *Country Centroids and Country Capitals*

Description

A `data.frame` with coordinates of country and province centroids and country capitals as reference for the `clean_coordinates`, `cc_cen` and `cc_cap` functions. Coordinates are based on the Central Intelligence Agency World Factbook `https://www.cia.gov/library/publications/the-world-factbook/` and `http://thematicmapping.org/downloads/world_borders.php`.

Format

A data frame with 5,142 observations on 10 variables. #'

iso3 ISO-3 code for each country, in case of provinces also referring to the country.

iso2 ISO-2 code for each country, in case of provinces also referring to the country.

name a factor; name of the country or province.

adm1_code adm code for countries and provinces

type identifying if the entry refers to a country or province level.

centroid.lon Longitude of the country centroid

centroid.lat Latitude of the country centroid

capital Name of the country capital, empty for provinces

capital.lon Longitude of the country capital

capital.lat Latitude of the country capital

Source

CENTRAL INTELLIGENCE AGENCY (2014) *The World Factbook*, Washington, DC.

https://www.cia.gov/library/publications/the-world-factbook/ http://thematicmapping.org/downloads/world_borders.php

Examples

```
data(countryref)
head(countryref)
```

---

institutions *Global Locations of Biodiversity Institutions*

Description

A global gazetteer for biodiversity institutions from various sources, including zoos, museums, botanical gardens, GBIF contributors, herbaria, university collections.

Format

A data frame with 12170 observations on 12 variables.

Source

Compiled from various sources:

- Global Biodiversity Information Facility www.gbif.org
- Wikipedia www.wikipedia.org
- Geonames www.geonames.org
- The Global Registry of Biodiversity Repositories www.grbio.org
- Index Herbariorum http://sweetgum.nybg.org/science/ih/
- Botanic Gardens Conservation International https://www.bgci.org/

Examples

```
data(institutions)
str(institutions)
```

pbdb_example *Example data from the Paleobiologydatabase*

Description

A dataset of 5000 flowering plant fossil occurrences as example for data of the paleobiology Database, downloaded using the paleobioDB packages as specified in the vignette "Cleaning_PBDB_fossils_with_Coordinate

Format

A data frame with 5000 observations on 36 variables.

Source

- The Paleobiology database https://paleobiodb.org/
- Sara Varela, Javier Gonzalez Hernandez and Luciano Fabris Sgarbi (2016). paleobioDB: Download and Process Data from the Paleobiology Database. R package version 0.5.0. https://CRAN.R-project.org/package=paleobioDB.

Examples

```
data(institutions)
str(institutions)
```

plot.spatialvalid *Plot Method for Class Spatialvalid*

Description

A set of plots to explore objects of the class `spatialvalid`. A plot to visualize the flags from clean_coordinates

Usage

```
## S3 method for class 'spatialvalid'
plot(x, lon = "decimallongitude",
  lat = "decimallatitude", bgmap = NULL, clean = TRUE,
  details = FALSE, pts_size = 1, font_size = 10, ...)
```

Arguments

x	an object of the class spatialvalid as from clean_coordinates.
lon	character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	character string. The column with the latitude coordinates. Default = "decimallatitude".
bgmap	an object of the class SpatialPolygonsDataFrame used as background map. Default = ggplot::borders()
clean	logical. If TRUE, non-flagged coordinates are included in the map.
details	logical. If TRUE, occurrences are color-coded by the type of flag.
pts_size	numeric. The point size for the plot.
font_size	numeric. The font size for the legend and axes
...	arguments to be passed to methods.

Value

A plot of the records flagged as potentially erroneous by clean_coordinates.

See Also

clean_coordinates

Examples

```
exmpl <- data.frame(species = sample(letters, size = 250, replace = TRUE),
                    decimallongitude = runif(250, min = 42, max = 51),
                    decimallatitude = runif(250, min = -26, max = -11))

test <- clean_coordinates(exmpl, species = "species",
                          tests = c("sea", "gbif", "zeros"),
                          verbose = FALSE)

summary(test)
plot(test)
```

---

write_pyrate *Create Input Files for PyRate*

Description

Creates the input necessary to run Pyrate, based on a data.frame with fossil ages (as derived e.g. from clean_fossils) and a vector of the extinction status for each sample. Creates files in the working directory!

Usage

```
write_pyrate(x, status, fname, taxon = "accepted_name",
  min_age = "min_ma", max_age = "max_ma", trait = NULL,
  path = getwd(), replicates = 1, cutoff = NULL, random = TRUE)
```

Arguments

x	data.frame. Containing fossil records with taxon names, ages, and geographic coordinates.
status	a vector of character strings of length nrow(x). Indicating for each record "extinct" or "extant".
fname	a character string. The prefix to use for the output files.
taxon	character string. The column with the taxon name. Default = "accepted_name".
min_age	character string. The column with the minimum age. Default = "min_ma".
max_age	character string. The column with the maximum age. Default = "max_ma".
trait	a numeric vector of length nrow(x). Indicating trait values for each record. Optional. Default = NULL.
path	a character string. giving the absolute path to write the output files. Default is the working directory.
replicates	a numerical. The number of replicates for the randomized age generation. See details. Default = 1.
cutoff	a numerical. Specify a threshold to exclude fossil occurrences with a high temporal uncertainty, i.e. with a wide temporal range between min_age and max_age. Examples: cutoff=NULL (default; all occurrences are kept in the data set) cutoff=5 (all occurrences with a temporal range of 5 Myr or higher are excluded from the data set)
random	logical. Specify whether to take a random age (between MinT and MaxT) for each occurrence or the midpoint age. Note that this option defaults to TRUE if several replicates are generated (i.e. replicates > 1). Examples: random = TRUE (default) random = FALSE (use midpoint ages)

Details

The replicate option allows the user to generate several replicates of the data set in a single input file, each time re-drawing the ages of the occurrences at random from uniform distributions with boundaries MinT and MaxT. The replicates can be analysed in different runs (see PyRate command -j) and combining the results of these replicates is a way to account for the uncertainty of the true ages of the fossil occurrences. Examples: replicates=1 (default, generates 1 data set), replicates=10 (generates 10 random replicates of the data set).

Value

PyRate input files in the working directory.

Note

See https://github.com/dsilvestro/PyRate/wiki for more details and tutorials on PyRate and PyRate input.

Examples

```
minages <- runif(250, 0, 65)
exmpl <- data.frame(accepted_name = sample(letters, size = 250, replace = TRUE),
                    lng = runif(250, min = 42, max = 51),
                    lat = runif(250, min = -26, max = -11),
                    min_ma = minages,
                    max_ma = minages + runif(250, 0.1, 65))

#a vector with the status for each record,
#make sure species are only classified as either extinct or extant,
#otherwise the function will drop an error

status <- sample(c("extinct", "extant"), size = nrow(exmpl), replace = TRUE)

#or from a list of species
status <- sample(c("extinct", "extant"), size = length(letters), replace = TRUE)
names(status) <- letters
status <- status[exmpl$accepted_name]

## Not run:
write_pyrate(x = exmpl,fname = "test", status = status)

## End(Not run)
```

Index