Package 'CoordinateCleaner'

November 9, 2017

Type Package

Title Automated Cleaning of Occurrence Records from Biological Collections

Version 1.0-1

Date 2017-11-08

Description Automated cleaning of geographic species occurrence records by automated flagging of problems common to biodiversity data from biological collections. Includes automated tests to easily flag (and exclude) records assigned to country or province centroids, the open ocean, the GBIF headquarters, urban areas or the location of biodiversity institutions (museums, zoos, botanical gardens, universities). Furthermore identifies per species outliers, zero coordinates, identical latitude/longitude and invalid coordinates. Also implements an algorithm to identify datasets with a significant proportion of rounded coordinates. Especially suited for large datasets. See <https://github.com/azizka/CoordinateCleaner/wiki> for more details and tutorials.

License GPL-3

Depends R (>= 3.0.0), sp

Imports geosphere, ggplot2, methods, raster, rgeos, rnaturalearth, stats

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Alexander Zizka [aut, cre], Daniele Silvestro [ctb]

Maintainer Alexander Zizka <alexander.zizka@bioenv.gu.se>

R topics documented:

`CoordinateCleaner-package`

Automated Cleaning of Occurrence Records from Biological Collections

---

Description

Automated cleaning of geographic species occurrence records by automated flagging of problems common to biodiversity data from biological collections. Includes automated tests to easily flag (and exclude) records assigned to country or province centroids, the open ocean, the GBIF headquarters, urban areas or the location of biodiversity institutions (museums, zoos, botanical gardens, universities). Furthermore identifies per species outliers, zero coordinates, identical latitude/longitude and invalid coordinates. Also implements an algorithm to identify datasets with a significant proportion of rounded coordinates. Especially suited for large datasets. See <https://github.com/azizka/CoordinateCleaner/wiki> for more details and tutorials.

Details

The DESCRIPTION file:

Package:	CoordinateCleaner
Type:	Package
Title:	Automated Cleaning of Occurrence Records from Biological Collections
Version:	1.0-1
Date:	2017-11-08
Authors@R:	c(person(given = "Alexander", family = "Zizka", email = "alexander.zizka@bioenv.gu.se", role = c("aut"
Description:	Automated cleaning of geographic species occurrence records by automated flagging of problems comm
License:	GPL-3
Depends:	R (>= 3.0.0), sp
Imports:	geosphere, ggplot2, methods, raster, rgeos, rnaturalearth, stats
LazyData:	true
RoxygenNote:	6.0.1
Author:	Alexander Zizka [aut, cre], Daniele Silvestro [ctb]
Maintainer:	Alexander Zizka <alexander.zizka@bioenv.gu.se>

Index of help topics:

Author(s)

NA

Maintainer: NA

`capitals` *Global Capital Locations*

Description

A gazetteer of global capital coordinates.

Usage

```
data("capitals")
```

Format

A data frame with 225 observations on the following 4 variables.

`ISO3` a factor, ISO-3 country code.

`capital` a factor, capital names.

`longitude` a numeric vector.

`latitude` a numeric vector.

Source

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

https://www.cia.gov/library/publications/the-world-factbook/

Examples

```
data(capitals)
str(capitals)
```

---

cc_cap *Flag Coordinates in Vicinity of Country Capitals.*

Description

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

Usage

```
cc_cap(x, lon = "decimallongitude", lat = "decimallatitude",
       buffer = 0.1, ref = NULL, value = "clean", verbose = TRUE)
```

Arguments

`x`	a data.frame. Containing geographical coordinates and species names.
`lon`	a character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	a character string. The column with the longitude coordinates. Default = "decimallatitude".
`buffer`	The buffer around each capital coordinate (the centre of the city), where records should be flagged as problematic, in decimal degrees. Default = 0.1.
`ref`	a SpatialPointsDataframe. Providing the geographic gazetteer. Can be any SpatialPointsDataframe, but the structure must be identical to `capitals`. Default = `capitals`
`value`	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
```

---

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

Description

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

Usage

```
cc_cen(x, lon = "decimallongitude", lat = "decimallatitude",
       buffer = 0.1, test = "both", ref = NULL, value = "clean", verbose = TRUE)
```

Arguments

`x`	a data.frame. Containing geographical coordinates and species names.
`lon`	a character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	a character string. The column with the longitude coordinates. Default = "decimallatitude".
`buffer`	numerical. The buffer around each province or country centroid, where records should be flagged as problematic, in decimal degrees. Default = 0.1.
`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`	a SpatialPointsDataframe. Providing the geographic gazetteer. Can be any SpatialPointsDataframe, but the structure must be identical to `centroids`. Default = `centroids`
`value`	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

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

cc_cen(x)
cc_cen(x, value = "flags")
```

---

cc_coun *Flag Coordinates Outside their Reported Country*

Description

Identifies 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	a data.frame. Containing geographical coordinates and species names, and a country assignment.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
iso3	a character string. The column with the country assignment of each record in three letter ISO code. Default = "countrycode".
ref	a SpatialPolygonDataframe. Providing the geographic gazetteer. Can be any SpatialPolygonsDataframe, but the structure must be identical to `countryref`. Default = `countryref`
value	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

Non-terrestrial records are ignored. Use `cc_sea` to flag these. See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")#non-terrestrial records are not flagged! Use cc_sea for these

## End(Not run)
```

---

cc_dupl *Flag Duplicated Records*

Description

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 = T)
```

Arguments

`x`	a data.frame. Containing geographical coordinates and species names.
`lon`	a character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	a character string. The column with the longitude 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`	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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

---

cc_equ *Flag Records with Identical lat/lon*

Description

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 = T)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude 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	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
```

---

cc_gbif *Flag Records Assigned to GBIF Headquarters*

Description

Flags records within 0.5 degree radius around the GBIF headquarters in Copenhagen, DK.

Usage

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

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
value	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
```

`cc_inst` *Flag Records in the Vicinity of Biodiversity Institutions*

Description

Flag 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",
        buffer = 0.001, ref = NULL, value = "clean", verbose = TRUE)
```

Arguments

`x`	a data.frame. Containing geographical coordinates and species names.
`lon`	a character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	a character string. The column with the longitude coordinates. Default = "decimallatitude".
`buffer`	numerical. The buffer around each province or country centroid, where records should be flagged as problematic, in decimal degrees. Default = 0.001 (= ~ 100m).
`ref`	a SpatialPointsDataframe. Providing the geographic gazetteer. Can be any SpatialPointsDataframe, but the structure must be identical to `institutions`. Default = `institutions`
`value`	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

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

cc_inst(x, buffer = 5)#large buffer for demonstration
cc_inst(x, value = "flags", buffer = 5)
```

---

cc_outl *Flag Geographic Outliers in Species Distributions*

Description

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 = 3, tdi = 1000, value = "clean", verbose = T)
```

Arguments

`x`	a data.frame. Containing geographical coordinates and species names.
`lon`	a character string. The column with the longitude coordinates. Default = "decimallongitude".
`lat`	a character string. The column with the longitude coordinates. Default = "decimallatitude".
`species`	a character string. The column with the species name. Default = "species".
`method`	a character string. Defining the method for outlier selection. See details. One of "distance", "quantile", "mad". Deafault = "quantile".
`mltpl`	numeric. The multiplier of the interquartile range (`method == 'quantile'`) or meadian absolute deviation (`method == 'mad'`)to identify outliers. See details. Default = 3.
`tdi`	numeric. The minimum absolute distance (`method == 'distance'`) of a record to all other records of a species to be indentified as outlier, in km. See details. Default = 1000.
`value`	a 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 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 medain 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`.

Value

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

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
cc_outl(x, method = "distance", value = "flags", tdi = 10000)
cc_outl(x, method = "distance", value = "flags", tdi = 1000)
```

---

cc_sea *Flag Non-terrestrial Coordinates*

Description

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, value = "clean", verbose = TRUE)
```

Arguments

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

lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
ref	a SpatialPointsPolygonsframe. Providing the geographic gazetteer. Can be any SpatialPolygonsDataframe, but the structure must be identical to landmass. See details. Default = landmass
value	a 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

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.: https://github.com/azizka/CoordinateCleaner/tree/master/extra_gazetteers.

Value

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

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

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

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

---

cc_urb *Flag Records Inside Urban Areas*

Description

Flags records from inside urban areas, based on a geographic gazetteer. Often records from large databases spann subtantial 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 = T)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
ref	a SpatialPolygonDataframe. Providing the geographic gazetteer with the urban areas. See details.
value	a 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

No default reference is provided with the package due to the large file size of such (global) gazetteers. You can download an example here: https://github.com/azizka/CoordinateCleaner/blob/master/extra_gazetteers/urbanareas.rda. Can be any SpatialPolygonDataframe, but the structure must be identical to urbanareas.

Value

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

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

```
## Not run:
# load reference
#See details section on where to download the reference data
load("extra_gazetteers/urbanareas.rda")

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

cc_urb(x, ref = urbanareas)
cc_urb(x, value = "flags", ref = urbanareas)

## End(Not run)
```

cc_val *Check Coordinate Validity in lat/lon*

Description

Checks if all coordinates in a data set are valid in a latitude/longitude coordinate reference system. That is non-numeric, not available coordinates and lat >90, la <-90, lon > 180 and lon < -180 are flagged.

Usage

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

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
value	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
```

cc_zero *Flag Zero Coordinates*

Description

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 = T)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude coordinates. Default = "decimallatitude".
buffer	numerical. The buffer around each province or country centroid, where records should be flagged as problematic, in decimal degrees. Default = 0.1.
value	a 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, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean".

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

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 = "flags")
```

centroids *Global Country and Province Centroids*

Description

A gazetteer of country and province centroids.

Usage

```
data("centroids")
```

Format

A data frame with 5142 observations on the following 6 variables.

`adm1_code` a factor; province code.

`iso3` a factor; country ISO-3 code.

`name` a factor; name of the country or province.

`type` a character vector; country or province.

`longitude` a numeric vector

`latitude` a numeric vector

Source

http://www.naturalearthdata.com/

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

https://www.cia.gov/library/publications/the-world-factbook/fields/2011.html

Examples

```
data(centroids)
str(centroids)
```

CleanCoordinates *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

```
CleanCoordinates(x, lon = "decimallongitude", lat = "decimallatitude",
                 species = "species", countries = NULL,
                 capitals = T, centroids = T,
                 countrycheck = F, duplicates = F, equal = T,
                 GBIF = T, institutions = T, outliers = F, seas = T,
                 urban = F, zeros = T,
                 capitals.rad = 0.05, centroids.rad = 0.01,
                 centroids.detail = "both", inst.rad = 0.001,
                 outliers.method = "quantile", outliers.mtp = 3,
                 outliers.td = 1000, zeros.rad = 0.5,
                 capitals.ref, centroids.ref, country.ref,
                 inst.ref, seas.ref, urban.ref,
                 value = "spatialvalid", verbose = T,
                 report = F)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude 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. A vector of the same length as rows in x, with country information for each record in ISO3 format. If missing, the countries test is skipped.
capitals	logical. If TRUE, tests a radius around adm-0 capitals. The radius is `capitals.rad`. Default = TRUE.
centroids	logical. If TRUE, tests a radius around country centroids. The radius is `centroids.rad`. Default = TRUE.
countrycheck	logical. If TRUE, tests if coordinates are from the country indicated in the country column. Default = FALSE.
duplicates	logical. If TRUE, 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. Default = FALSE.
equal	logical. If TRUE, tests for equal absolute longitude and latitude. Default = TRUE.
GBIF	logical. If TRUE, tests a one-degree radius around the GBIF headquarters in Copenhagen, Denmark. Default = TRUE.
institutions	logical. If TRUE, tests a radius around known biodiveristy institutions from `instiutions`. The radius is `inst.rad`. Default = TRUE.
outliers	logical. If TRUE, 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`). Default = TRUE.
seas	logical. If TRUE, tests if coordinates fall into the ocean. Default = TRUE.

urban	logical. If TRUE, tests if coordinates are from urban areas. Default = FALSE.
zeros	logical. If TRUE, tests for plain zeros, equal latitude and longitude and a radius around the point 0/0. The radius is zeros.rad. Default = TRUE.
capitals.rad	numeric. The radius around capital coordinates in degrees. Default = 0.1.
centroids.rad	numeric. The side length of the rectangle around country centroids in degrees. Default = 0.01.
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 biodiveristy institutions coordinates in degrees. Default = 0.001.
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 = 3.
outliers.td	numeric. The minimum distance of a record to all other records of a species to be indentified as outlier, in km. Default = 1000.
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 capitals dataset is used. Alternatives must be identical in structure.
centroids.ref	a data.frame with alternative reference data for the centroid test. If missing, the centroids dataset is used. Alternatives must be identical in structure.
country.ref	a SpatialPolygonsDataFrame as alternative reference for the countrycheck test. If missing, the rnaturalearth:ne_countries('medium') dataset is used.
inst.ref	a data.frame with alternative reference data for the biodiversity institution test. If missing, the institutions dataset is used. Alternatives must be identical in structure.
seas.ref	a SpatialPolygonsDataFrame as alternative reference for the seas test. If missing, the landmass dataset is used.
urban.ref	a SpatialPolygonsDataFrame as alternative reference for the urban test. If missing, 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. A reference gazetteer for the urban test is available at at https://github.com/azizka/CoordinateCleaner/extra_gazetteers. 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 medain 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`

Value

Depending on the output argument:

"spatialvalid" an object of calss `spatialvalid` with one column for each test. TRUE = clean coordinate, FALSE = potentially problematic coordinates. The summary column is FALSE if any test flagged the respective coordinate.

"flags" 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).

"cleaned" a `data.frame` of cleaned coordinates if `species = NULL` or a `data.frame` with cleaned coordinates and species ID otherwise

Note

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

See https://github.com/azizka/speciesgeocodeR/wiki for more details and tutorials.

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 <- CleanCoordinates(x = exmpl)

summary(test)
```

CleanCoordinatesDS *Geographic Coordinate Cleaning based on Dataset Properties*

Description

Identifies potentially problematic coordinates based on dataset properties. Includes test to flag 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.

Usage

```
CleanCoordinatesDS(x, lon = "decimallongitude", lat = "decimallatitude",
                   ds = "dataset",
                   ddmm = TRUE, periodicity = TRUE,
                   ddmm.pvalue = 0.025, ddmm.diff = 0.2,
                   periodicity.target = "lon_lat", periodicity.thresh = 3.5,
                   periodicity.diagnostics = FALSE,
                   periodicity.subsampling = NULL,
                   value = "dataset", verbose = TRUE)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude 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".
ddmm	logical. If TRUE, testing for erroneous conversion from a degree minute format (ddmm) to a decimal degree (dd.dd) format. See details.
periodicity	logical. If TRUE, testing for periodicity in the data, which can indicate imprecise coordinates, due to rounding or rasterization.
ddmm.pvalue	numeric. The p-value for the one-sided t-test to flag the ddmm test as passed or not. Both ddmm.pvalue and ddmm.diff must be met. Default = 0.025.
ddmm.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 0.025. Default = 0.2
periodicity.target	a character string. One of 'lat', 'lon', 'lon_lat'. Sets the target for the periodicity test which runs on latitude and longitude separately. If 'lon_lat'. Tests run sequentially and results for both and a combined flag are returned.
periodicity.thresh	numerical. The threshold to for flagging in the periodicity test. Indicates the factor by which one of the two periodic bins must outnumber the other. Default = 1.5. Higher values are more conservative/ flag less datasets.
periodicity.diagnostics	logical. If TRUE, plots a series of diagnostics visualizing the periodicity test. Default = FALSE.
periodicity.subsampling	numerical. If defined, only a random subsample of n = subsampling records is used for the periodicity test. Speeds up analyses, for the use with many large datasets.
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.

Details

This function checks the statistical distribution of 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 then expected have decimals blow 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 poison 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, either a summary per dataset `dataset`, a dataframe containing the records considered correct by the test ("clean") or a logical vector, with TRUE = test passed and FALSE = test failed/potentially problematic ("flags"). Default = "clean". If "dataset": `data.frame` with one row for each dataset in `x` and columns depending on the output option: 'detail' shows most level of detail, 'flag' shows only flags from the test and 'minimal' shows only the combined flags. Available columns are: `binomial.pvalue` = p-value compared to ddmm.pvalue; `perc.difference` = the percentage of difference from the expectation under a binomial test; `pass.ddmm` = logical flag summarizing the ddmm test, if TRUE: passed, if FALSE: potentially problematic; `mle` = the maximum likelihood for the rate parameters of the periodicity test; rate.ratio = rate ratio between the two rates of the periodicity model compared to `periodicity.thresh`; `zero.mle` = size of the maximum likelihood zero size bin from the zero test; `zero.rate.ratio` = ratio by which the number of zero decimals surpasses the number of records with other decimals; `pass.zero` = logical flag summarizing the zero test, if TRUE: passed, if FALSE: potentially problematic; `pass.periodicity` = logical flag summarizing the periodicity test, if TRUE: passed, if FALSE: potentially problematic. Flags for the periodicity test will be marked dependent on the meridional direction tested: 'lon' = longitude, 'lat' = latitude, 'com' = AND combination of the two former.

See Also

`CleanCoordinates`

Examples

```
#Create test dataset
clean <- data.frame(dataset = rep("clean", 1000),
                    decimallongitude = runif(min = -42, max = -40, n = 1000),
                    decimallatitude = runif(min = -12, 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 CleanCoordinatesDS
flags <- CleanCoordinatesDS(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)
```

countryref *Country Centroids and Country Capitals*

Description

A data.frame with coordinates of country centroids and country capitals as reference for the CleanCoordinates function. Coordinates are based on the Central Intelligence Agency World Factbook as provided at http://opengeocode.org/download/cow.php.

Usage

```
data("countryref")
```

Format

A data frame with 249 observations on 7 variables.

Source

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

http://opengeocode.org/download/cow.php

Examples

```
data(countryref)
```

dc_ddmm *Flag Datasets with a Degree Conversion Error*

Description

This test identifies 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 delimitor.

Usage

```
dc_ddmm(x, lon = "decimallongitude", lat = "decimallatitude", ds = "dataset",
        pvalue = 0.025, diff = 0.1, mat.size = 1000,
        value = "clean", verbose = TRUE)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude 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 = 0.1
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 smaler datasets, therefor, default = 1000. FOr large datasets try 10000.
value	a 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

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. See https://github.com/azizka/CoordinateCleaner/wiki/3.-Identifying-problematic-data-sets:-CleanCoordinatesDS for explanation and simulation results.

Value

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

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

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

dc_ddmm(x = clean, value = "flags")

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

prob <-  data.frame(species = letters[1:10],
                decimallongitude = lon,
```

```
                decimallatitude = lat,
                dataset = "FR")

dc_ddmm(x = prob, value = "flags")
```

---

dc_round *Flag Datasets with a Significant Fraction of Rounded Coordinates*

Description

Uses a three-rate Poisson process to model the distribution of coordinate decimals in the coordinates and to identify datasets with a significant fraction of records with low precision. Often these records have been subject to strong decimal rounding, or are based on rasterized data collection schemes.

Usage

```
dc_round(x, lon = "decimallongitude", lat = "decimallatitude",
         ds = "dataset", target = "lon_lat", threshold = 3.5,
         subsampling = NULL, diagnostics = FALSE, value = "clean", verbose = T)
```

Arguments

x	a data.frame. Containing geographical coordinates and species names.
lon	a character string. The column with the longitude coordinates. Default = "decimallongitude".
lat	a character string. The column with the longitude 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".
target	a character string. Defining the target of the test. One of "lon", "lat", "lon_lat", "lon_lat" is recommended. Default = "lon_lat".
threshold	numerical. Indicates the factor by which one of the two periodic bins must outnumber the other. Default = 1.5. Higher values are more conservative/flag less datasets.
subsampling	numeric. If NULL, the entire dataset is tested, if not NULL a random subsample of size subsampling is tested. This is recommedned for very large datasets, but subsampling values below 1000 are not recommended. Default = NULL
diagnostics	logical. If TRUE, plots a series of diagnostics visualizing the periodicity test. Default = FALSE.
value	a character string. Defining the output value. See value. Default = "clean".
verbose	logical. If TRUE reports the name of the test and the number of records flagged.

Details

To detect these patterns, we model the distribution of decimals in the range [0,1] as the result of a 3-rate Poisson process, where the first rate (lambda_0) is assigned to the range $R_0 = [0, t_0)]$ and the second and third rates (lambda_1 and lambda_2) are assigned to successive bins of sizes s_1 and s_2$. We use maximum likelihood to estimate the three rates and the sizes of the bins. The number of resulting bins (N) depends on the quantities t_0,s_1,s_2, all of which are constrained

to be positive and smaller than 0.3. We note that, while the number of bins changes based on their size, the number of parameters in the model is constant and equal to 6. We expect this model to return high values of lambda_0 compared to lambda_1, ambda_2, and t_0 to be small, in the presence of a bias increasing the frequency of 0s in the decimals. Periodic biases are expected to result in strongly different rates in the following bins (e.g. lambda_1 » lambda_2) and small estimated values of s_1,s_2 such that they allow for multiple repeated peaks. After empirically inspecting the behaviour of these estimates and their ability to detect biases in the data we defined arbitrary thresholds to flag a data set as potentially biased by poor precision

Value

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

Note

See https://github.com/azizka/CoordinateCleaner/wiki for more details and tutorials.

Examples

```
clean <- data.frame(species = letters[1:10],
                decimallongitude = runif(100, -180, 180),
                decimallatitude = runif(100, -90,90),
                dataset = "clean")
#biased dataset
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(species = letters[1:10],
                   decimallongitude = bias.long,
                   decimallatitude = bias.lat,
                   dataset = "rounded")
test <- rbind(clean, bias)

## Not run:
#run CleanCoordinatesDS
flags <- CleanCoordinatesDS(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)
```

institutions *Global Locations of Biodiversity Institutions*

Description

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

Usage

```
data("institutions")
```

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 Registriy of Biodiversity Repositories www.grbio.org
- Index Herbariorum http://sciweb.nybg.org/Science2/IndexHerbariorum.asp
- Botanic Gardens Conservation International https://www.bgci.org/

Examples

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

landmass *Global Coastlines*

Description

A SpatialPolygonsDataFrame with global coastlines.

Usage

```
data("landmass")
```

Note

Most of the times it might be desirable to only flag records far away from the coast as problematic rather than thos close to the coastline (which might be due to disagreeements in coastlines, or low gps uncertainty). For these cases, there is a alternative coastline reference buffered by one degree available at https://github.com/azizka/CoordinateCleaner/tree/master/extra_gazetteers.

Source

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

Examples

```
data("landmass")
```

`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 CleanCoordinates

Usage

```
## S3 method for class 'spatialvalid'
plot(x, bgmap = NULL, clean = T, details = T,
              pts.size = 1, font.size = 10, ...)
```

Arguments

`x`	an object of the class `spatialvalid` as from `CleanCoordinates`.
`bgmap`	an object of the class `SpatialPolygonsDataFrame` used as background map. Default = `landmass`
`clean`	logical. If TRUE, non-flagged coordinates are included in the map.
`details`	logical. If TRUE, occurences are colourcoded 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
`...`	additional arguments passed to other methods

Value

A plot of the records flagged as potentially erroneous by `CleanCoordinates`.

See Also

`CleanCoordinates`

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 <- CleanCoordinates(exmpl, species = "species", verbose = FALSE)

summary(test)
```

Index