Package ‘lingtypology’

October 13, 2022

Type Package

Title Linguistic Typology and Mapping

Version 1.1.9

Depends R (>= 3.5.0)

Imports leaflet, leaflet.minicharts, stats, utils, stringdist, grDevices, jsonlite

Description Provides R with the Glottolog database <https://glottolog.org/> and some more abilities for purposes of linguistic mapping. The Glottolog database contains the catalogue of languages of the world. This package helps researchers to make a linguistic maps, using philosophy of the Cross-Linguistic Linked Data project <https://clld.org/>, which allows for while at the same time facilitating uniform access to the data across publications. A tutorial for this package is available on GitHub pages <https://docs.ropensci.org/lingtypology/> and package vignette. Maps created by this package can be used both for the investigation and linguistic teaching. In addition, package provides an ability to download data from typological databases such as WALS, AUTOTYP and some others and to create your own database website.

License GPL (>= 2)

URL https://CRAN.R-project.org/package=lingtypology,
 https://github.com/ropensci/lingtypology/,
 https://ropensci.github.io/lingtypology/

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

LazyData TRUE

RoxygenNote 7.1.2

Encoding UTF-8

Suggests knitr, rmarkdown, testthat, covr, MASS, sp, rgeos, rgdal, ggplot2, ape

VignetteBuilder knitr

NeedsCompilation no
Author  George Moroz [aut, cre] (<https://orcid.org/0000-0003-1990-6083>),
        Kirill Koncha [ctb] (<https://orcid.org/0000-0003-0676-2658>),
        Mikhail Leonov [ctb],
        Anna Smirnova [ctb],
        Sasha Shakhnova [ctb]
Maintainer  George Moroz <agricolamz@gmail.com>
Repository  CRAN
Date/Publication  2022-06-24 12:10:02 UTC

R topics documented:

abvd ........................................... 3
abvd.feature .................................. 4
afbo.feature .................................. 4
aff.lang ...................................... 5
amap .......................................... 6
area.lang ..................................... 6
atlas.database ................................ 7
autotyp ....................................... 7
autotyp.feature ................................ 8
bantu .......................................... 9
bantu.feature .................................. 9
bivaltyp.feature ................................ 10
circassian .................................... 10
countries ..................................... 11
country.lang ................................... 11
eurasianphonology ................................ 12
eurasianphonology.feature ..................... 13
ggmap.feature .................................. 13
glottolog .................................... 15
gltc.iso ....................................... 16
gltc.lang ...................................... 16
imports ....................................... 17
is.glottolog .................................. 17
iso.gltc ....................................... 18
iso.lang ....................................... 19
lang.aff ...................................... 19
lang.country ................................... 20
lang.gltc ...................................... 21
lang.iso ....................................... 21
lat.lang ....................................... 22
level.lang .................................... 23
long.lang ..................................... 23
map.feature .................................... 24
oto_mangueanIC ................................ 30
oto_mangueanIC.feature ....................... 30
phoible ...................................... 31
**abvd**

ABVD’s Language identifiers

**Description**

Language identifiers from ABVD (https://abvd.eva.mpg.de/austronesian/). This dataset is created for `abvd.feature` function.

**Usage**

`abvd`

**Format**

A data frame with 1468 rows and 2 variables:

- **id** language identifier
- **glottocode** Glottocode
*afbo.feature*

---

**Description**

This function downloads data from ABVD ([https://abvd.eva.mpg.de/austronesian/](https://abvd.eva.mpg.de/austronesian/)) and changes language names to the names from lingtypology database. You need the internet connection.

**Usage**

```r
abvd.feature(feature)
```

**Arguments**

- `feature` A character vector that define a language id from ABVD (e.g. "1", "292").

**Author(s)**

George Moroz <agricolamz@gmail.com>

**See Also**


**Examples**

```r
# abvd.feature(c(292, ?))
```

---

**Description**

This function downloads data from AfBo ([https://afbo.info/](https://afbo.info/)) and changes language names to the names from lingtypology database. You need the internet connection.

**Usage**

```r
afbo.feature(features = "all", na.rm = TRUE)
```
**Arguments**

- **features**: A character vector that define with an affix functions from AfBo (e.g. "all", "adjectivizer", "focus").
- **na.rm**: Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

**See Also**


**Examples**

```r
# afbo.feature()
# afbo.feature(c("adjectivizer", "adverbializer"))
```

---

**aff.lang**

*Get affiliation by language*

**Description**

Takes any vector of languages and returns affiliation.

**Usage**

```r
aff.lang(x)
```

**Arguments**

- **x**: A character vector of the languages (can be written in lower case)

**Author(s)**

George Moroz <agricolamz@gmail.com>

**See Also**

- `area.lang`, `country.lang`, `gltc.lang`, `iso.lang`, `lat.lang`, `long.lang`, `subc.lang`, `url.lang`

**Examples**

```r
aff.lang('Korean')
aff.lang(c('Korean', 'Polish'))
```
area.lang

Description
Takes any vector of languages and returns macro area.

Usage
area.lang(x)

Arguments
x character vector of the languages (can be written in lower case)

Author(s)
George Moroz <agricolamz@gmail.com>

See Also
aff.lang, country.lang, gltc.lang, iso.lang, lat.lang, long.lang, subc.lang, url.lang

Examples
area.lang('Adyghe')
area.lang(c('Adyghe', 'Aduge'))
**atlas.database**  
Create an atlas

**Description**
This function creates an rmarkdown based atlas from data provided by users. This function creates the template, after it should be rendered by rmarkdown package. The DT package is required during the rendering.

**Usage**
```r
atlas.database(
    languages,
    latitude,
    longitude,
    features,
    atlas.name = "",
    author = ""
)
```

**Arguments**
- `languages`: character vector of languages (can be written in lower case)
- `latitude`: numeric vector of latitudes (optional)
- `longitude`: numeric vector of longitudes (optional)
- `features`: dataframe where each column is a feature set
- `atlas.name`: string with an atlas name
- `author`: string with the authors list

**autotyp**  
**AUTOTYP’s Language identifiers**

**Description**
Language identifiers from AUTOTYP v. 0.1.4 ([https://github.com/autotyp/autotyp-data/](https://github.com/autotyp/autotyp-data/)). This dataset is created for `autotyp.feature` function.

**Usage**
```
autotyp
```

**Format**
An object of class tbl_df (inherits from tbl, data.frame) with 1336 rows and 3 columns.
Details

```#' @format A data frame with 1336 rows and 3 variables:

  path  path to the dataset in autotyp repo
  variable  variable name
  file  topic name```

Description

This function downloads data from AUTOTYP (https://github.com/autotyp/autotyp-data#the-autotyp-database) and changes language names to the names from lingtypology database. You need the internet connection.

Usage

```autotyp.feature(features, na.rm = TRUE)```

Arguments

- `features`: A character vector that define with a feature names from AUTOTYP.
- `na.rm`: Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

See Also


```abvd.feature, afbo.feature, oto_mangueanIC.feature, phoible.feature, sails.feature, uralex.feature, valpal.feature, wals.feature```

Examples

```# autotyp.feature(c('Has Gender', 'Has Numeral Classifiers'))```
**bantu**

---

**BANTU’s Language identifiers**

**Description**

Language identifiers from BANTU (https://abvd.eva.mpg.de/bantu/index.php). This dataset is created for `bantu.feature` function.

**Usage**

```
bantu
```

**Format**

A data frame with 430 rows and 2 variables:

- `id` BANTU word id
- `word` word

---

**bantu.feature**  

**Download BANTU data**

**Description**

This function downloads data from Bantu Basic Vocabulary Database (https://abvd.eva.mpg.de/bantu/index.php) and changes language names to the names from lingtypology database. You need the internet connection.

**Usage**

```
bantu.feature(features)
```

**Arguments**

- `features` A character vector that define with a feature ids from BANTU ('house', 'cat').

**Author(s)**

Anna Smirnova <annedadaa@gmail.com>

**See Also**

`abvd.feature`, `afbo.feature`, `autotyp.feature`, `oto_mangueanIC.feature`, `phoible.feature`, `sails.feature`, `uralex.feature`, `valpal.feature`

**Examples**

```
# bantu.feature(c('house', 'cat'))
```
bivaltyp.feature  Download BivalTyp data

Description
This function downloads data from BivalTyp (https://www.bivaltyp.info/) and changes language names to the names from lingtypology database. You need the internet connection.

Usage
bivaltyp.feature()

Author(s)
George Moroz <agricolamz@gmail.com>

See Also
abvd.feature, afbo.feature, autotyp.feature, oto_mangueanIC.feature, phoible.feature, sails.feature, valpal.feature, wals.feature

circassian  Circassian villages in Russia

Description
A dataset contains the list of the Circassian villages in Russia with genealogical affiliation, coordinates and district names. Most data collected during the fieldworks (2011–2018).

Usage
circassian

Format
A data frame with 158 rows and 6 variables:

longitude  longitude
latitude  latitude
village  name of the village
countries

dialect names of the Circassian dialects
language according standard Circassian devision there are Adyghe and Kabardian languages

countries

Catalogue of countries

Description

Catalogue of countries, ISO-codes and official languages

Usage
countries

Format

A data frame with 189 rows and 5 variables:

alpha3 ISO 3166-3 code of the country
alpha2 ISO 3166-2 code of the country
country_name Country name
additional_names Additional names of the country
official_languages Official languages

country.lang

Get country by language

Description

Takes any vector of languages and returns countries where those languages are used as ISO 3166-1 alpha-2 codes.

Usage
country.lang(x, full_name = TRUE)

Arguments

x A character vector of the languages (can be written in lower case)
full_name A logical value, whether return ISO 3166-2 codes or full names.
Author(s)
George Moroz <agricolamz@gmail.com>

See Also
aff.lang, area.lang, gltc.lang, iso.lang, lat.lang, long.lang, subc.lang, url.lang

Examples

country.lang('Korean')
country.lang(c('Korean', 'Polish'))

data frame with 19825 rows and 19 variables:
id Language id
iso ISO code
name Another language name
type Language or dialect
language Language name
latitude latitude
longitude longitude
gen1 Language Family
gen2 Language Family
tones Inventory of tones
syllab Syllab structure
cluster Cluster
finals Finals
source Source
comment Comment

data frame with 19825 rows and 19 variables:
id Language id
iso ISO code
name Another language name
type Language or dialect
language Language name
latitude latitude
longitude longitude
gen1 Language Family
gen2 Language Family
tones Inventory of tones
syllab Syllab structure
cluster Cluster
finals Finals
source Source
comment Comment
eurasianphonology.feature

- **contr**: Contributor
- **segment_type**: Vowels or consonants
- **segments**: Segments
- **glottocode**: Glottocode

---

**eurasianphonology.feature**

*Opens data from the database of Eurasian phonological inventories*

---

**Description**

This function opens downloaded data from the database of Eurasian phonological inventories ([https://eurphon.info](https://eurphon.info)).

**Usage**

`eurasianphonology.feature()`

**Author(s)**

Kirill Koncha <majortomblog@gmail.com>

**See Also**


**Examples**

`eurasianphonology.feature()`

---

**ggmap.feature**

*Create a map with ggplot2*

---

**Description**

Map a set of languages and color them by feature.
Usage

ggmap.feature(
  languages,
  features = "",
  latitude = NA,
  longitude = NA,
  color = NULL,
  title = NULL,
  legend = TRUE,
  width = 2,
  opacity = 1,
  map.orientation = "Atlantic"
)

Arguments

languages character vector of languages (can be written in lower case).
features character vector of features.
latitude numeric vector of latitudes.
longitude numeric vector of longitudes.
color vector of colors or palette. The color argument can be (1) a character vector of RGM or named colors; (2) the name of an RColorBrewer palette; (3) the full name of a viridis palette; (4) a function that receives a single value between 0 and 1 and returns a color. For more examples see colorNumeric.
title title of a legend.
legend logical. If TRUE, function show legend. By default is TRUE.
width a numeric vector of radius for circles or width for barcharts in minicharts.
opacity a numeric vector of marker opacity.
map.orientation a character vector with values "Pacific" and "Atlantic". It distinguishes Pacific-centered and Atlantic-centered maps. By default is "Atlantic".

Author(s)

George Moroz <agricolamz@gmail.com>

Examples

ggmap.feature(c("Adyghe", "Russian"))
Description

A dataset contains the original catalogue of languages of the world involving genealogical affiliation, macro-area, country, iso code, and coordinates.

Usage

glottolog

Format

A data frame with 26285 rows and 10 variables:

- **glottocode**: languoid code from Glottolog 4.5
- **language**: name of the language
- **iso**: code based on ISO 639–3 [https://iso639-3.sil.org/](https://iso639-3.sil.org/)
- **level**: languoid type: dialect or language (possible values are dialect, language, family, bookkeeping, pseudo family, sign language, unclassifiable, pidgin, unattested, artificial language, speech register, mixed language)
- **area**: have six values Africa, Australia, Eurasia, North America, Papunesia, South America
- **latitude**: latitude
- **longitude**: longitude
- **countries**: list of countries, where the language is spoken
- **affiliation**: genealogical affiliation
- **subclassification**: subclassification in a Newick format

Details


Source

[https://glottolog.org/](https://glottolog.org/)
gltc.iso

Get Glottocode by ISO 639–3 code

Description


Usage

`gltc.iso(x)`

Arguments

- `x`  A character vector of the Glottocodes.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

`aff.lang, area.lang, lat.lang, long.lang`

Examples

- `gltc.iso('ady')`
- `gltc.iso(c('ady', 'rus'))`

---

gltc.lang

Get Glottocode by language

Description

Takes any vector of languages and returns Glottocode.

Usage

`gltc.lang(x)`

Arguments

- `x`  A character vector of the languages (can be written in lower case)

Author(s)

George Moroz <agricolamz@gmail.com>
See Also

aff.lang, area.lang, country.lang, iso.lang, lat.lang, long.lang, subc.lang, url.lang

Examples

gltc.lang('Adyghe')
gltc.lang(c('Adyghe', 'Udi'))

is.glottolog(x, response = FALSE)

Arguments

x A character vector of languages (can be written in lower case) or ISO codes
response logical. If TRUE, when language is absent, return warnings with a possible candidates.

Author(s)

George Moroz <agricolamz@gmail.com>
Examples

is.glottolog(c('Adyghe', 'Russian'))
is.glottolog('Buyaka')

# Add warning message with suggestions
is.glottolog(c('Adygey', 'Russian'), response = TRUE)
# > FALSE TRUE
# Warning message:
# In is.glottolog(c('Adyge', 'Russian'), response = TRUE) :
# Language Adyge is absent in our version of the Glottolog database. Did you mean Aduge, Adyghe?

iso.gltc  

Get ISO 639–3 code by Glottocode

Description

Takes any vector of Glotocodes and returns ISO code.

Usage

iso.gltc(x)

Arguments

x  A character vector of Glotocodes.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, lat.lang, long.lang

Examples

iso.gltc('adyg1241')
is.gltc(c('adyg1241', 'udii1243'))
iso.lang  Get ISO 639–3 code by language

Description
Takes any vector of languages and returns ISO code.

Usage
iso.lang(x)

Arguments
x A character vector of the languages (can be written in lower case)

Author(s)
George Moroz <agricolamz@gmail.com>

See Also
aff.lang, area.lang, country.lang, gltc.lang, lat.lang, long.lang, subc.lang, url.lang

Examples
iso.lang('Adyghe')
iso.lang(c('Adyghe', 'Udi'))

lang.aff  Get languages by affiliation

Description
Takes any vector of affiliations and returns languages.

Usage
lang.aff(x, include.dialects = FALSE, list = FALSE)

Arguments
x A character vector of the affiliations (can be written in lower case)
include.dialects logical. If TRUE, it returns all languages and dialects, if FALSE it returns only languages.
list logical. If TRUE, it returns a list of languages, if FALSE it returns a named vector.
**lang.country**

**Author(s)**
George Moroz <agricolamz@gmail.com>

**See Also**

`lang.iso`

**Examples**

```r
lang.aff('Slavic')
lang.aff(c('Slavic', 'Celtic'))
lang.aff(c('Slavic', 'Celtic'), list = TRUE)
```

---

**Description**

Takes any vector of countries and returns languages.

**Usage**

```r
lang.country(x, list = TRUE)
```

**Arguments**

- `x` character vector of the countries (in alpha-2 ISO codes)
- `list` logical. If TRUE, it returns a list of languages, if FALSE it returns a named vector.

**Author(s)**
George Moroz <agricolamz@gmail.com>

**See Also**

`aff.lang`, `country.lang`, `gltc.lang`, `iso.lang`, `lat.lang`, `long.lang`, `subc.lang`, `url.lang`

**Examples**

```r
lang.country('AD')
lang.country(c('AD', 'AE'))
```
**lang.gltc**  *

*Takes any vector of Glottocodes and returns languages.*

**Usage**

```
lang.gltc(x)
```

**Arguments**

- `x` A character vector of the Glottocodes.

**Author(s)**

George Moroz <agricolamz@gmail.com>

**See Also**

`lang.aff`

**Examples**

```
lang.gltc('adyg1241')
lang.gltc(c('adyg1241', 'udii1243'))
```

---

**lang.iso**  *

*Takes any vector of ISO codes and returns languages.*

**Usage**

```
lang.iso(x)
```

**Arguments**

- `x` A character vector of the ISO codes.

**Author(s)**

George Moroz <agricolamz@gmail.com>
See Also

| lang.aff |

Examples

```r
lat.lang('Adyghe')
long.lang('Adyghe')
lat.lang(c('Adyghe', 'Russian'))
long.lang(c('Adyghe', 'Russian'))
```

---

**lat.lang**

*Get latitude by language*

**Description**

Takes any vector of languages and returns latitude.

**Usage**

```r
lat.lang(x)
```

**Arguments**

- `x` A character vector of the languages (can be written in lower case)

**Author(s)**

George Moroz <agricolamz@gmail.com>

**See Also**

```r
aff.lang, area.lang, country.lang, gltc.lang, iso.lang, long.lang, subc.lang, url.lang
```

**Examples**

```r
lat.lang('Adyghe')
long.lang('Adyghe')
lat.lang(c('Adyghe', 'Russian'))
long.lang(c('Adyghe', 'Russian'))
```
level.lang

Get a level of language by language

Description
Takes any vector of languages and returns a level of language.

Usage
level.lang(x)

Arguments
x character vector of the languages (can be written in lower case)

Author(s)
Sasha Shakhnova

See Also
aff.lang, country.lang, gltc.lang, iso.lang, lat.lang, long.lang, subc.lang, url.lang

Examples
level.lang('Russian Sign Language')
level.lang(c('Archi', 'Chechen'))

long.lang
Get longitude by language

Description
Takes any vector of languages and returns longitude.

Usage
long.lang(x, map.orientation = "Pacific")

Arguments
x A character vector of the languages (can be written in lower case)
map.orientation A character vector with values "Pacific" and "Atlantic". It distinguishes Pacific-centered and Atlantic-centered maps. By default is "Pacific".
Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, country.lang, gltc.lang, iso.lang, lat.lang, subc.lang, url.lang

Examples

lat.lang('Adyghe')
long.lang('Adyghe')
lat.lang(c('Adyghe', 'Russian'))
long.lang(c('Adyghe', 'Russian'))
long.lang(c('Adyghe', 'Aleut'), map.orientation = "Pacific")

Description

Map a set of languages and color them by feature or two sets of features.

Usage

map.feature(languages,
            features = "",
            label = "",
            popup = "",
            latitude = NA,
            longitude = NA,
            label.hide = TRUE,
            label.fsize = 15,
            label.font = "sans-serif",
            label.position = "right",
            label.emphasize = list(NULL, "black"),
            shape = NULL,
            shape.size = 20,
            pipe.data = NULL,
            shape.color = "black",
            stroke.features = NULL,
            point.cluster = FALSE,
            density.estimation = NULL,
            density.method = "fixed distance",
            density.estimation.color = NULL,
            density.estimation.opacity = 0.6,
            density.points = TRUE,
density.width = NULL,
density.legend = TRUE,
density.legend.opacity = 1,
density.legend.position = "bottomleft",
density.title = "",
density.control = FALSE,
isogloss = NULL,
isogloss.color = "black",
isogloss.opacity = 0.2,
isogloss.line.width = 3,
isogloss.width = NULL,
color = NULL,
stroke.color = NULL,
image.url = NULL,
image.width = 100,
image.height = 100,
image.X.shift = 0,
image.Y.shift = 0,
title = NULL,
stroke.title = NULL,
control = "",
legend = TRUE,
legend.opacity = 1,
legend.position = "topright",
stroke.legend = TRUE,
stroke.legend.opacity = 1,
stroke.legend.position = "bottomleft",
width = 5,
stroke.radius = 9.5,
opacity = 1,
stroke.opacity = 1,
scale.bar = TRUE,
scale.bar.position = "bottomleft",
minimap = FALSE,
minimap.position = "bottomright",
minimap.width = 150,
minimap.height = 150,
facet = NULL,
tile = "OpenStreetMap.Mapnik",
tile.name = NULL,
tile.opacity = 1,
zoom.control = FALSE,
zoom.level = NULL,
rectangle.lng = NULL,
rectangle.lat = NULL,
rectangle.color = "black",
line.lng = NULL,
line.lat = NULL,
line.type = "standard",
line.color = "black",
line.opacity = 0.8,
line.label = NULL,
line.width = 3,
graticule = NULL,
minichart = "bar",
minichart.data = NULL,
minichart.time = NULL,
minichart.labels = FALSE,
map.orientation = "Pacific",
radius = NULL
)

Arguments

languages character vector of languages (can be written in lower case)
features character vector of features
label character vector of strings that will appear near points
popup character vector of strings that will appear in pop-up window
latitude numeric vector of latitudes
longitude numeric vector of longitudes
label.hide logical. If FALSE, labels are displayed allways. If TRUE, labels are displayed on mouse over. By default is TRUE.
label.fsize numeric value of the label font size. By default is 14.
label.font string with values of generic family: "serif", "sans-serif", "monospace", or font name e. g. "Times New Roman"
label.position the position of labels: "left", "right", "top", "bottom"
label.emphasize is the list. First argument is a vector of points in dataframe that should be emphasized. Second argument is a string with a color for emphasis.
shape 1. if TRUE, creates icons (up to five categories) for values in the features variable;
2. it also could be a vector of any strings that represents the levels of the features variable;
3. it also could be a string vector that represents the number of observations in dataset.
shape.size size of the shape icons
pipe.data this variable is important, when you use map.feature with dplyr pipes. Expected usage: pipe.data = .
shape.color color of the shape icons
stroke.features additional independent stroke features
point.cluster logical. If TRUE, points will be united into clusters.
density.estimation
   additional independent features, used for density estimation

density.method
   string with one of the two methods: "kernal density estimation" or "fixed
distance" (default)

density.estimation.color
   vector of density polygons' colors

density.estimation.opacity
   a numeric vector of density polygons opacity.

density.points
   logical. If FALSE, it doesn’t show points in polygons.

density.width
   for density.method = "fixed distance" it is a numeric measure (1 is 1km). For
density.method = "kernal density estimation" it is a vector with two measures
(first is latitude, second is longitude). Defaults are normal reference bandwidth
(see bandwidth.nrd).

density.legend
   logical. If TRUE, function show legend for density features. By default is
   FALSE.

density.legend.opacity
   a numeric vector of density-legend opacity.

density.legend.position
   the position of the legend: "topright", "bottomright", "bottomleft", "topleft"

density.title
   title of a density-feature legend

density.control
   logical. If TRUE, function show layer control buttons for density plot. By de-
   fault is FALSE

isogloss
   dataframe with corresponding features

isogloss.color
   vector of isoglosses' colors

isogloss.opacity
   a numeric vector of density polygons opacity.

isogloss.line.width
   a numeric value for line width

isogloss.width
   for density.method = "fixed distance" it is a numeric measure (1 is 1km). For
density.method = "kernal density estimation" it is a vector with two measures
(first is latitude, second is longitude). Defaults are normal reference bandwidth
(see bandwidth.nrd).

color
   vector of colors or palette. The color argument can be (1) a character vector of
RGM or named colors; (2) the name of an RColorBrewer palette; (3) the full
name of a viridis palette; (4) a function that receives a single value between 0
and 1 and returns a color. For more examples see colorNumeric

stroke.color
   vector of stroke colors

image.url
   character vector of URLs with an images

image.width
   numeric vector of image widths

image.height
   numeric vector of image heights

image.X.shift
   numeric vector of image’s X axis shift relative to the latitude-longitude point

image.Y.shift
   numeric vector of image’s Y axis shift relative to the latitude-longitude point
title: title of a legend.
control: vector of grouping values that make it possible to create control panel that can turn off/on some points on the map.
legend: logical. If TRUE, function show legend. By default is TRUE.
legend.opacity: a numeric vector of legend opacity.
legend.position: the position of the legend: "topright", "bottomright", "bottomleft","topleft"
stroke.legend: logical. If TRUE, function show stroke.legend. By default is FALSE.
stroke.legend.opacity: a numeric vector of stroke.legend opacity.
stroke.legend.position: the position of the stroke.legend: "topright", "bottomright", "bottomleft","topleft"
width: a numeric vector of radius for circles or width for barcharts in minicharts.
stroke.radius: a numeric vector of stroke radii for the circles.
opacity: a numeric vector of marker opacity.
stroke.opacity: a numeric vector of stroke opacity.
scale.bar: logical. If TRUE, function shows scale-bar. By default is TRUE.
scale.bar.position: the position of the scale-bar: "topright", "bottomright", "bottomleft","topleft"
minimap: logical. If TRUE, function shows mini map. By default is FALSE.
minimap.position: the position of the minimap: "topright", "bottomright", "bottomleft","topleft"
minimap.width: The width of the minimap in pixels.
minimap.height: The height of the minimap in pixels.
facet: character vector that provide a grouping variable. If it is no NULL, then as a result a list of leaflets for sync or latticeView functions from mapview package is returned.
tile: a character vector with a map tiles, popularized by Google Maps. See here for the complete set.
tile.name: a character vector with a user’s map tiles’ names.
tile.opacity: numeric value from 0 to 1 denoting opacity of the tile.
zoom.control: logical. If TRUE, function shows zoom controls. By default is FALSE.
zoom.level: a numeric value of the zoom level.
rectangle.lng: vector of two longitude values for rectangle.
rectangle.lat: vector of two latitude values for rectangle.
rectangle.color: vector of rectangle border color.
line.lng: vector of two (or more) longitude values for line.
line.lat: vector of two (or more) latitude values for line.
map.feature

- **line.type**: a character string indicating which type of line is to be computed. One of "standard" (default), or "logit". The first one should be combined with the arguments line.lat and line.lng and provide simple lines. Other variant "logit" is the decision boundary of the logistic regression made using longitude and latitude coordinates (works only if feature argument have two levels).

- **line.color**: vector of line color.

- **line.opacity**: a numeric vector of line opacity.

- **line.label**: character vector that will appear near the line.

- **line.width**: a numeric vector of line width.

- **graticule**: a numeric vector for graticule spacing in map units between horizontal and vertical lines.

- **minichart**: citation from leaflet.minicharts package: "Possible values are "bar" for bar charts, "pie" for pie charts, "polar-area" and "polar-radius".".

- **minichart.data**: citation from leaflet.minicharts package: "A numeric matrix with number of rows equal to the number of elements in lng or lat and number of column equal to the number of variables to represent. If parameter time is set, the number of rows must be equal to the length of lng times the number of unique time steps in the data."

- **minichart.time**: citation from leaflet.minicharts package: "A vector with length equal to the number of rows in chartdata and containing either numbers representing time indices or dates or datetimes. Each unique value must appear as many times as the others. This parameter can be used when one wants to represent the evolution of some variables on a map."

- **minichart.labels**: citation from leaflet.minicharts package: "Should values be displayed above chart elements."

- **map.orientation**: a character vector with values "Pacific" and "Atlantic". It distinguishes Pacific-centered and Atlantic-centered maps. By default is "Pacific".

- **radius**: deprecated argument

**Author(s)**

George Moroz <agricolamz@gmail.com>

**Examples**

map.feature(c("Adyghe", "Russian"))
Description
Language identifiers from Oto-Manguean Inflectional Class Database (https://oto-manguean.surrey.ac.uk/). This dataset is created for oto_mangueanIC.function.

Usage
oto_mangueanIC

Format
An object of class tbl_df (inherits from tbl, data.frame) with 20 rows and 2 columns.

Details
#' @format A data frame with 20 rows and 2 variables:
Language.name Language names from Oto-Manguean Inflectional Class Database
language Language names from Glottolog database

oto_mangueanIC.feature
Download Oto-Manguean Inflectional Class Database data

Description
This function downloads data from Oto-Manguean Inflectional Class Database (https://oto-manguean.surrey.ac.uk/) and creates a language column with the names from lingtypology database. You need the internet connection.

Usage
oto_mangueanIC.feature()

Author(s)
George Moroz <agricolamz@gmail.com>

See Also
abvd.feature, afbo.feature, autotyp.feature, phoible.feature, sails.feature, uralex.feature, valpal.feature, wals.feature
**Description**

Language correspondencies for Phoible ([https://phoible.org/](https://phoible.org/)). This dataset is created for `phoible.feature` function.

**Usage**

```r
phoible
```

**Format**

A data frame with 2185 rows and 2 variables:

- `language` (language)
- `Glottocode` (Glottocode)

**Description**

This function downloads data from PHOIBLE ([https://phoible.org/](https://phoible.org/)) and changes language names to the names from lingtypology database. You need the internet connection.

**Usage**

```r
phoible.feature(source = "all", na.rm = TRUE)
```

**Arguments**

- **source**: A character vector that define with a source names from PHOIBLE (possible values: "all", "aa", "gm", "ph", "ra", "saphon", "spa", "upsid").
- **na.rm**: Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

**See Also**

- abvd.feature
- afbo.feature
- autotyp.feature
- bivaltyp.feature
- eurasianphonology.feature
- oto_mangueanIC.feature
- sails.feature
- soundcomparisons.feature
- uralex.feature
- valpal.feature
- vanuatu.feature
- wals.feature
- abvd.feature
- afbo.feature
- autotyp.feature
- oto_mangueanIC.feature
- sails.feature
- uralex.feature
- valpal.feature
- wals.feature
Examples

# phoible.feature()
# phoible.feature(c('consonants', 'vowels'), source = 'UPSID')

phonological_profiles  *Number of consonants and presence of ejectives*

Description

Number of consonants and presence of ejectives

Usage

phonological_profiles

Format

A data frame with 19 rows and 4 variables:

- **language**  language name
- **consonants**  number of consonants. Based on UPSID database.
- **vowels**  number of vowels. Based on UPSID database.
- **ejectives**  presence of ejective sounds.
- **tone**  presence of tone.
- **stress**  presence of stress.
- **long_vowels**  presence of long vowels.

polygon.points_fd  *Get poligons from fixed distance circles around coordinates*

Description

This function is based on this answer: https://www.r-bloggers.com/merging-spatial-buffers-in-r/

Usage

polygon.points_fd(latitude, longitude, width)

Arguments

- **latitude**  numeric vector of latitudes
- **longitude**  numeric vector of longitudes
- **width**  radius for creating poligons around points
**polygon.points_kde**  
*Get kernel density estimation polygon from coordinates*

**Description**

This function is based on this answer: https://gis.stackexchange.com/a/203623/

**Usage**

```r
polygon.points_kde(latitude, longitude, latitude.width, longitude.width)
```

**Arguments**

- `latitude`: numeric vector of latitudes
- `longitude`: numeric vector of longitudes
- `latitude.width`: bandwidths for latitude values. Defaults to normal reference bandwidth (see `bandwidth.nrd`).
- `longitude.width`: bandwidths for longitude values. Defaults to normal reference bandwidth (see `bandwidth.nrd`).

**providers**  
*Providers*

**Description**

List of all providers with their variations taken from leaflet package

**Usage**

```r
providers
```

**Format**

A list of characters

**Source**

https://github.com/leaflet-extras/leaflet-providers/blob/master/leaflet-providers.js
sails.feature  Download SAILS data

Description

This function downloads data from SAILS ([https://sails.cld.org/](https://sails.cld.org/)) and changes language names to the names from lingtypology database. You need the internet connection.

Usage

sails.feature(features, na.rm = TRUE)

Arguments

- **features**: A character vector that define with a feature ids from SAILS (e.g. "and1", "argex4-1-3").
- **na.rm**: Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

See Also


Examples

```r
# sails.feature(c("and1", "and11"))
```

soundcomparisons  SOUNDCOMPARISONS’s Language identifiers

Description

Language identifiers from SOUNDCOMPARISONS ([https://soundcomparisons.com/](https://soundcomparisons.com/)). This dataset is created for soundcomparisons.feature function.

Usage

soundcomparisons

Format

An object of class data.frame with 556 rows and 3 columns.
soundcomparisons.feature

Details

#' @format A data frame with 556 rows and 2 variables:

LanguageName  SOUNDCOMPARISONS language identifier
LanguageId     Language Id

Description

This function downloads data from SOUNDCOMPARISONS (https://soundcomparisons.com/) and changes language names to the names from lingtypology database. You need the internet connection.

Usage

soundcomparisons.feature(word)

Arguments

word A character vector that define with a feature ids from SOUNDCOMPARISONS (e.g. "one", "sharp_fem", "near_neut", "on_the_left", "I_will_give", "write_ipv_sg", "your_pl_pl").

Author(s)

Anna Smirnova

See Also


Examples

# soundcomparisons.feature(c("sun", "house"))
subc.lang  

Get subclassification by language

Description
Takes any vector of languoids and returns subclassification in the Newick tree format.

Usage
subc.lang(x)

Arguments
x A character vector of the languoids (can be written in lower case)

Author(s)
George Moroz <agricolamz@gmail.com>

See Also
aff.lang, area.lang, country.lang, gltc.lang, iso.lang, lat.lang, long.lang

Examples
subc.lang('Korean')
subc.lang(c('Korean', 'Lechitic'))

urate

UraLex’s Language identifiers

Description
Language identifiers from UraLex (https://github.com/lexibank/uralex/). This dataset is created for uralex.feature function.

Usage
urate

Format
A data frame with 27 rows and 3 variables:

language language name from database
Glottocode Glottocodes
language2 language from lingtypology
uralex.feature  Download UraLex data

Description
This function downloads data from UraLex (https://github.com/lexibank/uralex/) and changes language names to the names from lingtypology database. You need the internet connection.

Usage
uralex.feature(na.rm = TRUE)

Arguments

na.rm Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

Author(s)
George Moroz <agricolamz@gmail.com>

See Also

Examples
# uralex.feature()

url.lang  Make a url-link to glottolog page for a language

Description
Takes any vector of languages and returns links to glottolog pages.

Usage
url.lang(x, popup = "")

Arguments

x A character vector of languages (can be written in lower case)

popup character vector of strings that will appear in pop-up window of the function map.feature
Author(s)

George Moroz <agricolamz@gmail.com>

See Also

aff.lang, area.lang, country.lang, gltc.lang, iso.lang, lat.lang, long.lang, subc.lang

Examples

url.lang("Korean")
url.lang(c("Gangou", "Hachijo", "Adyghe", "Ganai"))

valpal.feature

Download ValPaL data

Description

This function downloads data from ValPal (https://valpal.info) and changes language names to the names from lingtypology database. You need the internet connection.

Usage

valpal.feature(na.rm = FALSE)

Arguments

na.rm Logical. If TRUE function removes all languages not available in lingtypology database. By default is FALSE.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also


Examples

# valpal.feature()
vanuatu.feature  Download Vanuatu Voices data

Description
This function downloads data from Vanuatu Voices (https://vanuatuvoices.clld.org/). You need the internet connection.

Usage
vanuatu.feature(features, na.rm = TRUE)

Arguments
features  A vector with parameters from Concepts (https://vanuatuvoices.clld.org/parameters)
na.rm    Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

Author(s)
Mikhail Leonov

See Also

wals  WALS's Language identifiers

Description
Language identifiers from WALS (https://wals.info/). This dataset is created for wals.feature function.

Usage
wals

Format
A data frame with 2950 rows and 2 variables:
wals.code  WALS language identifier
glottocode  Glottocode
wals.feature  Download WALS data

Description
This function downloads data from WALS (https://wals.info/) and changes language names to the names from lingtypology database. You need the internet connection.

Usage
wals.feature(features, na.rm = TRUE)

Arguments
- features: A character vector that define with a feature ids from WALS (e.g. "1a", "21b").
- na.rm: Logical. If TRUE function removes all languages not available in lingtypology database. By default is TRUE.

Author(s)
George Moroz <agricolamz@gmail.com>

See Also

Examples
# wals.feature(c("1a", "20a"))
Index

* datasets
  abvd, 3
  amap, 6
  autotyp, 7
  bantu, 9
  circassian, 10
  countries, 11
  eurasianphonology, 12
  glottolog, 15
  oto_mangueanIC, 30
  phoible, 31
  phonological_profiles, 32
  providers, 33
  soundcomparisons, 34
  uralex, 36
  wals, 39
%>%(imports), 17
%>%, 17

abvd, 3
abvd.feature, 3, 4, 5, 8–10, 13, 30, 31, 34, 35, 37–40
afbo.feature, 4, 4, 8–10, 13, 30, 31, 34, 35, 37–40
aff.lang, 5, 6, 12, 16–20, 22–24, 36, 38
amap, 6
area.lang, 5, 6, 12, 16–19, 22, 24, 36, 38
atlas.database, 7
autotyp, 7
autotyp.feature, 4, 5, 7, 8, 9, 10, 13, 30, 31, 34, 35, 37–40

bandwidth.nrd, 27, 33
bantu, 9
bantu.feature, 9, 9
bivaltyp.feature, 4, 5, 8, 10, 13, 30, 31, 34, 35, 37–40

circassian, 10
colorNumeric, 14, 27
countries, 11
country.lang, 5, 6, 11, 17, 19, 20, 22–24, 36, 38
eurasianphonology, 12
eurasianphonology.feature, 4, 5, 8, 10, 12, 13, 30, 31, 34, 35, 37–40
ggmap.feature, 13
glottolog, 15
gltc.lang, 5, 6, 12, 16, 19, 20, 22–24, 36, 38
imports, 17
is.glottolog, 17
iso.gltc, 18
iso.lang, 5, 6, 12, 17, 19, 20, 22–24, 36, 38
lang.aff, 19, 21, 22
lang.country, 20
lang.gltc, 21
lang.iso, 20, 21
lat.lang, 5, 6, 12, 16–20, 22, 23, 24, 36, 38
level.lang, 23
long.lang, 5, 6, 12, 16–20, 22, 23, 23, 36, 38
map.feature, 24
oto_mangueanIC, 30
oto_mangueanIC.feature, 4, 5, 8–10, 13, 30, 31, 34, 35, 37–40
phoible, 31
phoible.feature, 4, 5, 8–10, 13, 30, 31, 34, 35, 37–40
phonological_profiles, 32
polygon.points_fd, 32
polygon.points_kde, 33
providers, 33
sails.feature, 4, 5, 8–10, 13, 30, 31, 34, 35, 37–40
soundcomparisons, 34
soundcomparisons.feature, 4, 5, 8, 10, 13,
30, 31, 34, 35, 37–40
subc.lang, 5, 6, 12, 17, 19, 20, 22–24, 36, 38
uralex, 36
uralex.feature, 4, 5, 8–10, 13, 30, 31,
34–36, 37, 38–40
url.lang, 5, 6, 12, 17, 19, 20, 22–24, 37
valpal.feature, 4, 5, 8–10, 13, 30, 31, 34,
35, 37, 38, 39, 40
vanuatu.feature, 4, 5, 8, 10, 13, 30, 31, 34,
35, 37, 38, 39, 40
wals, 39
wals.feature, 4, 5, 8, 10, 13, 30, 31, 34, 35,
37–39, 40