Description Provides R with the Glottolog database <http://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 <http://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 7.1.0

Encoding UTF-8

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

VignetteBuilder knitr

NeedsCompilation no
R topics documented:

- abvd
- abvd.feature
- afbo.feature
- aff.lang
- amap
- area.lang
- atlas.database
- autotyp
- autotyp.feature
- circassian
- ggmap.feature
- glottolog
- gltc.iso
- gltc.lang
- imports
- is.glottolog
- iso.gltc
- iso.lang
- lang.aff
- lang.gltc
- lang.iso
- lat.lang
- long.lang
- map.feature
- oto_mangueanIC
- oto_mangueanIC.feature
- phoible
- phoible.feature
- phonological_profiles
- polygon.points_fd
- polygon.points_kde
- providers
- sails.feature
- uralex
- uralex.feature
- uralex.feature
- url.lang
- valpal.feature
- wals
- wals.feature

Index 33
abvd

**ABVD’s Language identifiers**

**Description**

Language identifiers from ABVD ([https://abvd.shh.mpg.de/austronesian/](https://abvd.shh.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

**abvd.feature**

**Download ABVD data**

**Description**

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

**Usage**

`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**

`afbo.feature`, `autotyp.feature`, `oto_mangueanIC.feature`, `phoible.feature`, `sails.feature`, `valpal.feature`, `wals.feature`

**Examples**

```r
# abvd.feature(c(292, ?))
```
afbo.feature  Download AfBo data

Description

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

Usage

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.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

abvd.feature, autotyp.feature, oto_mangueanIC.feature, phoible.feature, sails.feature, uralex.feature, valpal.feature, wals.feature

Examples

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

aff.lang  Get affiliation by language

Description

Takes any vector of languages and return affiliation.

Usage

aff.lang(x)

Arguments

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

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

See Also
area.lang, iso.lang, lat.lang, long.lang

Examples
aff.lang("Korean")
aff.lang(c("Korean", "Polish"))

amap

Atlantic center template for ggmap.feature() function

Description
.

Usage
amap

Format
A list with 9 variables.

area.lang

Get macro area by language

Description
Takes any vector of languages and return 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>
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

See Also

`aff.lang`, `iso.lang`, `lat.lang`, `long.lang`

Examples

```r
area.lang('Adyghe')
area.lang(c('Adyghe', 'Aduge'))
```
**autotyp**

### AUTOTYP’s Language identifiers

**Description**

Language identifiers from AUTOTYP v. 0.1.0 ([https://github.com/autotyp/autotyp-data](https://github.com/autotyp/autotyp-data)). This dataset is created for `autotyp.feature` function.

**Usage**

```r
autotyp
```

**Format**

An object of class `data.frame` with 2950 rows and 2 columns.

**Details**

- `LID` language identifier
- `Glottocode` Glottocode

---

**autotyp.feature**

**Download AUTOTYP data**

**Description**

This function downloads data from AUTOTYP ([https://github.com/autotyp/autotyp-data#the-autotyp-database](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**

```r
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.

**Author(s)**

George Moroz <agricolamz@gmail.com>
See Also

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

Examples

# autotyp.feature(c('Gender', 'Numeral classifiers'))

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
- **dialect**  names of the Circassian dialects
- **language**  according standard Circassian division there are Adyghe and Kabardian languages
ggmap.feature

Create a map with ggplot2

Description

Map a set of languages and color them by feature.

Usage

```r
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"))

---

glottolog

Catalogue of languages of the world

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 20290 rows and 8 variables:

- **affiliation**: genealogical affiliation
- **glottocode**: languoid code from Glottolog 4.1
- **language**: name of the language
- **level**: languoid type: dialect or language
- **longitude**: longitude
- **latitude**: latitude
- **area**: have six values Africa, Australia, Eurasia, North America, Papunesia, South America

Details


Source

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>
is.glottolog

See Also

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

Examples

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

---

is.glottolog

**Objects imported from other packages**

---

**Descriptions**

**is.glottolog**

Are these languages in glottolog?

**Description**

Takes any vector of languages or ISO codes and return a logical vector.

**Usage**

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

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

```r
# 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 Glottocodes and returns ISO code.

#### Usage

```r
iso.gltc(x)
```

#### Arguments

- `x` A character vector of Glottocodes.

#### Author(s)

George Moroz <agricolamz@gmail.com>

#### See Also

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

#### Examples

```r
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, lat.lang, long.lang

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

---

**lang.aff**  
*Get languages by affiliation*

**Description**  
Takes any vector of affiliations and return 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.
Author(s)

George Moroz <agricolamz@gmail.com>

See Also

lang.iso

Examples

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

Description

Takes any vector of Glottocodes and return 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**  
*Get language by ISO 639–3 code*

**Description**  
Takes any vector of ISO codes and return 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
lang.iso('ady')
lang.iso(c('ady', 'rus'))
```

---

**lat.lang**  
*Get latitude by language*

**Description**  
Takes any vector of languages and return latitude.

**Usage**  
`lat.lang(x)`

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

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

See Also
aff.lang, area.lang, iso.lang, long.lang

Examples

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

long.lang Get longitude by language

Description
Takes any vector of languages and return 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, iso.lang, lat.lang

Examples

```r
l.lat.lang('Adyghe')
long.lang('Adyghe')
l.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,
map.feature

image.height = 100,
image.X.shift = 0,
image.Y.shift = 0,
title = NULL,
stroke.title = NULL,
control = FALSE,
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,
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 polygones.
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, secnd 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.
map.feature

- **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 default 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.
- **stroke.title**: title of a stroke-feature legend.
- **control**: logical. If TRUE, function show layer control buttons. By default is FALSE
- **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.
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.
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."
oto_mangueanIC

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"))

---

<table>
<thead>
<tr>
<th>oto_mangueanIC</th>
<th>Oto-Manguean Inflectional Class Database Language identifiers</th>
</tr>
</thead>
</table>

Description

Language identifiers from Oto-Manguean Inflectional Class Database (http://wals.info/). This dataset is created for oto_mangueanIC.feature 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 (http://www.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

Examples
# oto_mangueanIC.feature()

phoible

Phoible glottolog - language correspondencies

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

Usage
phoible

Format
A data frame with 2185 rows and 2 variables:
language  language
Glottocode  Glottocode
phoible.feature  Download PHOIBLE data

Description

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

Usage

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.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

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 polygons 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 poligon from coordinates*

**Description**

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

**Usage**

```
polygon.points_kde(latitude, longitude, latitude.width, longitude.width)
```
**providers**

### 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`).

### Description

List of all providers with their variations taken from leaflet package

### Usage

```
providers
```

### Format

A list of characters

### Source


---

**sails.feature**  
**Download SAILS data**

### Description

This function downloads data from SAILS (http://sails.cll.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.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

abvd.feature, afbo.feature, autotyp.feature, oto_mangueanIC.feature, phoible.feature, uralex.feature, valpal.feature, wals.feature

Examples

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

---

uralex  UraLex’s Language identifiers

Description

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

Usage

uralex

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

abvd.feature, afbo.feature, autotyp.feature, oto_mangueanIC.feature, sails.feature, uralex.feature, valpal.feature, wals.feature

Examples

# uralex.feature()

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

Description

Takes any vector of languages and return 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>

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

Description
This function downloads data from ValPal (http://www.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()
wals

**WALS’s Language identifiers**

**Description**

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

**Usage**

wals

**Format**

An object of class `tbl_df` (inherits from `tbl.data.frame`) with 2679 rows and 2 columns.

**Details**

```
# 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 (http://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

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

Examples

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

*Topic datasets
  abvd, 3
  amap, 5
  autotyp, 7
  circassian, 8
  glottolog, 10
  oto_mangueanIC, 23
  phoible, 24
  phonological_profiles, 25
  providers, 27
  uralex, 28
  wals, 31
  %>% (imports), 12
  %>%, 12
  abvd, 3
  abvd.feature, 3, 3, 4, 8, 24, 25, 28–30, 32
  afbo.feature, 3, 4, 8, 24, 25, 28–30, 32
  aff.lang, 4, 6, 11–14, 17
  amap, 5
  area.lang, 5, 5, 11–14, 17
  atlas.database, 6
  autotyp, 7
  autotyp.feature, 3, 4, 7, 24, 25, 28–30, 32
  bandwidth.nrd, 20, 21, 27
  circassian, 8
  colorNumeric, 9, 21
  ggmap.feature, 9
  glottolog, 10
  gltc.iso, 11
  gltc.lang, 11
  imports, 12
  is.glottolog, 12
  iso.gltc, 13
  iso.lang, 5, 6, 14, 17
  lang.aff, 14, 15, 16
  lang.gltc, 15
  lang.iso, 15, 16
  lat.lang, 5, 6, 11–14, 16, 17
  long.lang, 5, 6, 11–14, 17, 17
  map.feature, 18
  oto_mangueanIC, 23
  oto_mangueanIC.feature, 3, 4, 8, 23, 24, 25, 28–30, 32
  phoible, 24
  phoible.feature, 3, 4, 8, 24, 25, 28, 30, 32
  phonological_profiles, 25
  polygon.points_fd, 26
  polygon.points_kde, 26
  providers, 27
  sails.feature, 3, 4, 8, 24, 25, 27, 29, 30, 32
  uralex, 28
  uralex.feature, 4, 8, 24, 25, 28, 29, 30, 32
  url.lang, 29
  valpal.feature, 3, 4, 8, 24, 25, 28, 29, 30, 32
  wals, 31
  wals.feature, 3, 4, 8, 24, 25, 28–31, 31