

Package ‘lingtypology’

May 28, 2025

Type Package

Title Linguistic Typology and Mapping

Version 1.1.21

Depends R (>= 4.1.0)

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 7.3.2.9000

Encoding UTF-8

Suggests knitr, rmarkdown, testthat, MASS, sp, sf, ape

VignetteBuilder knitr

NeedsCompilation no

Author George Moroz [aut, cre] (ORCID: <https://orcid.org/0000-0003-1990-6083>),
 Kirill Koncha [ctb] (ORCID: <https://orcid.org/0000-0003-0676-2658>),
 Mikhail Leonov [ctb],
 Anna Smirnova [ctb],
 Ekaterina Zalivina [ctb]

Maintainer George Moroz <agricolamz@gmail.com>

Repository CRAN

Date/Publication 2025-05-27 22:10:02 UTC

Contents

abvd	3
abvd.feature	4
afbo.feature	4
aff.lang	5
area.lang	6
atlas.database	6
autotyp	7
autotyp.feature	8
bantu	8
bantu.feature	9
bivalentyp.feature	10
circassian	10
countries	11
country.lang	11
eurasianphonology	12
eurasianphonology.feature	13
frequency_list.feature	13
glottolog	14
gltc.iso	15
gltc.lang	16
grambank.feature	16
imports	17
is.glottolog	17
iso.gltc	18
iso.lang	19
iso3.iso1	19
iso_639	20
lang.aff	21
lang.country	21
lang.gltc	22
lang.iso	23
lat.lang	23
level.lang	24
long.lang	25
map.feature	25

oto_mangueanIC	31
oto_mangueanIC.feature	32
phoible	32
phoible.feature	33
phonological_profiles	33
polygon.points_fd	34
polygon.points_kde	34
providers	35
sails.feature	35
soundcomparisons	36
soundcomparisons.feature	37
subc.lang	37
uralex	38
uralex.feature	39
url.lang	39
valpal.feature	40
vanuatu.feature	41
wals	41
wals.feature	42
Index	43

abvd	<i>ABVD's Language identifiers</i>
------	------------------------------------

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

abvd.feature

Download ABVD data

Description

This function downloads data from ABVD (<https://abvd.eva.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](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

Examples

```
# abvd.feature(c(292, 7))
```

afbo.feature

Download AfBo data

Description

This function downloads data from AfBo (<https://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.

See Also

[abvd.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

[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	<i>Get affiliation by language</i>
----------	------------------------------------

Description

Takes any vector of languages and returns affiliation.

Usage

```
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

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

area.lang	<i>Get macro area by language</i>
-----------	-----------------------------------

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('Kabardian')
area.lang(c('Kabardian', 'Aduge'))
```

atlas.database	<i>Create an atlas</i>
----------------	------------------------

Description

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

Usage

```
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. 1.1.1 (<https://github.com/autotyp/autotyp-data/>). This dataset is created for `autotyp.feature` function.

Usage

```
autotyp
```

Format

An object of class `data.frame` with 1342 rows and 3 columns.

Details

#' @format A data frame with 1342 rows and 3 variables:

path path to the dataset in autotyp repo

variable variable name

file topic name

autotyp.feature	<i>Download AUTOTYP data</i>
-----------------	------------------------------

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](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

[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	<i>BANTU's Language identifiers</i>
-------	-------------------------------------

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	<i>Download BANTU data</i>
---------------	----------------------------

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	<i>Download BivalTyp data</i>
------------------	-------------------------------

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](#)

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#) # [bivaltyp.feature\(\)](#)

circassian	<i>Circassian villages in Russia</i>
------------	--------------------------------------

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

district names of the subjects of the Russian Federation: kbr — Kabardino–Balkar Republic, kch — Karachay–Cherkess Republic, kk — Krasnodar Krai, ra — Republic of Adygea, stv — Stavropol Krai

dialect names of the Circassian dialects

language according standard Circassian devision there are West Circassian 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'))
```

eurasianphonology	<i>Eurasianphonology data</i>
-------------------	-------------------------------

Description

Data from The database of Eurasian phonological inventories (<https://eurphon.info>). This dataset is created for [eurasianphonology.feature](#) function.

Usage

```
eurasianphonology
```

Format

A 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

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>).

Usage

```
eurasianphonology.feature()
```

Author(s)

Kirill Koncha <majortomblog@gmail.com>

See Also

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [oto_mangueanIC.feature](#),
[phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#),
[vanuatu.feature](#), [wals.feature](#)

Examples

```
eurasianphonology.feature()
```

frequency_list.feature

Download frequency list

Description

This function downloads frequency list from OpenSubtitles2018 (<https://opus.nlpl.eu/>). You need the internet connection.

Usage

```
frequency_list.feature(languages, list_type = "full")
```

Arguments

languages	ISO 639-1 language code and some others ('ze_en', 'ze_zh', 'zh_cn', 'zh_tw', 'pt_br'). Possible values: 'af', 'ar', 'bg', 'bn', 'br', 'bs', 'ca', 'cs', 'da', 'de', 'el', 'en', 'eo', 'es', 'et', 'eu', 'fa', 'fi', 'fr', 'gl', 'he', 'hi', 'hr', 'hu', 'hy', 'id', 'is', 'it', 'ja', 'ka', 'kk', 'ko', 'lt', 'lv', 'mk', 'ml', 'ms', 'nl', 'no', 'pl', 'pt', 'pt_br', 'ro', 'ru', 'si', 'sk', 'sl', 'sq', 'sr', 'sv', 'ta', 'te', 'tl', 'tr', 'uk', 'ur', 'vi', 'ze_en', 'ze_zh', 'zh_cn', 'zh_tw'.
list_type	Type of frequency list. Possible values: 'full', '50k', 'ignored'. By default is full.

Author(s)

Ekaterina Zalivina <zalivina01@mail.ru>

See Also

[abvd.feature](#), [afbo.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

[abvd.feature](#), [afbo.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [uralex.feature](#), [valpal.feature](#), [wals.feature](#)

Examples

```
# frequency_list.feature('ro')
# frequency_list.feature('en', '50k')
# frequency_list.feature(c('en', 'ru'), '50k')
```

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 27034 rows and 10 variables:

glottocode languoid code from Glottolog 5.2

language name of the language

iso code based on ISO 639-3 <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

Hammarstrom, Harald and Forkel, Robert and Haspelmath, Martin and Bank, Sebastian. 2025. Glottolog 5.2. Leipzig: Max Planck Institute for Evolutionary Anthropology. <https://doi.org/10.5281/zenodo.10804357> (Available online at <http://glottolog.org>, Accessed on 2024-03-12.)

Source

<https://glottolog.org/>

gltc.iso

Get Glottocode by ISO 639–3 code

Description

Takes any vector of ISO 639–3 codes and returns Glottocodes.

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	<i>Get Glottocode by language</i>
-----------	-----------------------------------

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('Kabardian')
gltc.lang(c('Kabardian', 'Udi'))
```

grambank.feature	<i>Download Grambank data</i>
------------------	-------------------------------

Description

This function downloads data from Grambank (<https://grambank.clld.org/>) and changes language names to the names from lingtypology database. You need the internet connection.

Usage

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

Arguments

features	A character vector that define with a feature ids from Grambank (e. g. "gb026", "gb042").
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](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

Examples

```
# grambank.feature(c("gb026", "gb042"))
```

imports	<i>Objects imported from other packages</i>
---------	---

Description

These objects are imported from other packages. Follow the links to their documentation.

magrittr [%>%](#)

is.glottolog	<i>Are these languages in glottolog?</i>
--------------	--

Description

Takes any vector of languages or ISO codes and returns 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

```

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

## Not run:
# Add warning message with suggestions
is.glottolog(c('Adyge', 'Russian'), response = TRUE)
# > FALSE TRUE
# Warning message:
# In is.glottolog(c('Kabardia', 'Russian'), response = TRUE) :
# Language Kabardia is absent in our version of the Glottolog database.
# Did you mean Kabardian, Greater Kabardian?

## End(Not run)

```

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 Glottocodes.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

[aff.lang](#), [area.lang](#), [lat.lang](#), [long.lang](#)

Examples

```

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

```

iso.lang	<i>Get ISO 639-3 code by language</i>
----------	---------------------------------------

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('Kabardian')
iso.lang(c('Kabardian', 'Udi'))
```

iso3.iso1	<i>Get ISO 639-3 code from ISO 639-1</i>
-----------	--

Description

Takes any vector of ISO 639-1 codes and returns ISO 639-3 code.

Usage

```
iso3.iso1(x)
```

Arguments

x	A character vector of ISO 639-3 codes.
---	--

Author(s)

Ekaterina Zalivina <zalivina01@mail.ru>

See Also

[aff.lang](#), [area.lang](#), [lat.lang](#), [long.lang](#)

Examples

```
iso3.iso1('bs')
iso3.iso1(c('co', 'it', 'ar'))
```

iso_639

ISO 639-3 is a set of codes that defines three-letter identifiers for all known human languages.

Description

ISO 639 provides three language code sets: one is a two-letter code (ISO 639-1) and two others are three-letter codes (ISO 639-2 and ISO 639-3) for the representation of names of languages. ISO 639-1 was devised primarily for use in terminology, lexicography and linguistics. ISO 639-2 was devised primarily for use in terminology and bibliography. ISO 639-3 was devised to provide a comprehensive set of identifiers for all languages for use in a wide range of applications, including linguistics, lexicography and internationalization of information systems. It attempts to represent all known full languages.

Usage

iso_639

Format

A data frame with 188 rows and 5 variables:

ISO_639_3 The three-letter 639-3 identifier

ISO_639_2_B Equivalent 639-2 identifier of the bibliographic applications code set

ISO_639_2_T Equivalent 639-2 identifier of the terminology applications code set

ISO_639_1 Equivalent 639-1 identifier

Ref_Name Reference language name

Details

(Available online at <https://iso639-3.sil.org/>, Accessed on 2022-05-23.)

Source

<https://iso639-3.sil.org/>

lang.aff	<i>Get languages by affiliation</i>
----------	-------------------------------------

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.

Author(s)

George Moroz <agricolamz@gmail.com>

See Also

[lang.iso](#)

Examples

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

lang.country	<i>Get language by country</i>
--------------	--------------------------------

Description

Takes any vector of countries and returns languages.

Usage

```
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

```
lang.country('AD')
lang.country(c('AD', 'AE'))
```

lang.gltc

Get language by Glottocode

Description

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	<i>Get language by ISO 639–3 code</i>
----------	---------------------------------------

Description

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

```
lang.iso('ady')  
lang.iso(c('ady', 'rus'))
```

lat.lang	<i>Get latitude by language</i>
----------	---------------------------------

Description

Takes any vector of languages and returns 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>

See Also

[aff.lang](#), [area.lang](#), [country.lang](#), [gltc.lang](#), [iso.lang](#), [long.lang](#), [subc.lang](#), [url.lang](#)

Examples

```
lat.lang('Kabardian')
long.lang('Kabardian')
lat.lang(c('Kabardian', 'Russian'))
long.lang(c('Kabardian', '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	<i>Get longitude by language</i>
-----------	----------------------------------

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

map.feature	<i>Create a map</i>
-------------	---------------------

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.estation = NULL,  
  density.method = "fixed distance",  
  density.estation.color = NULL,  
  density.estation.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: "kernel 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 = "kernel 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 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 = "kernel 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	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.

<code>minimap.position</code>	the position of the minimap: "topright", "bottomright", "bottomleft", "topleft"
<code>minimap.width</code>	The width of the minimap in pixels.
<code>minimap.height</code>	The height of the minimap in pixels.
<code>facet</code>	character vector that provide a grouping variable. If it is no NULL, then as a result a list of leaflets for <code>sync</code> or <code>latticeView</code> functions from <code>mapview</code> package is returned.
<code>tile</code>	a character vector with a map tiles, popularized by Google Maps. See here for the complete set.
<code>tile.name</code>	a character vector with a user's map tiles' names.
<code>tile.opacity</code>	numeric value from 0 to 1 denoting opacity of the tile.
<code>zoom.control</code>	logical. If TRUE, function shows zoom controls. By default is FALSE.
<code>zoom.level</code>	a numeric value of the zoom level.
<code>rectangle.lng</code>	vector of two longitude values for rectangle.
<code>rectangle.lat</code>	vector of two latitude values for rectangle.
<code>rectangle.color</code>	vector of rectangle border color.
<code>line.lng</code>	vector of two (or more) longitude values for line.
<code>line.lat</code>	vector of two (or more) latitude values for line.
<code>line.type</code>	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 <code>line.lat</code> and <code>line.lng</code> 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).
<code>line.color</code>	vector of line color.
<code>line.opacity</code>	a numeric vector of line opacity.
<code>line.label</code>	character vector that will appear near the line.
<code>line.width</code>	a numeric vector of line width.
<code>graticule</code>	a numeric vector for graticule spacing in map units between horizontal and vertical lines.
<code>minichart</code>	citation from <code>leaflet.minicharts</code> package: "Possible values are "bar" for bar charts, "pie" for pie charts, "polar-area" and "polar-radius"."
<code>minichart.data</code>	citation from <code>leaflet.minicharts</code> package: "A numeric matrix with number of rows equal to the number of elements in <code>lng</code> or <code>lat</code> and number of column equal to the number of variables to represent. If parameter <code>time</code> is set, the number of rows must be equal to the length of <code>lng</code> times the number of unique time steps in the data."
<code>minichart.time</code>	citation from <code>leaflet.minicharts</code> package: "A vector with length equal to the number of rows in <code>chartdata</code> 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."

<code>minichart.labels</code>	citation from leaflet.minicharts package: "Should values be displayed above chart elements."
<code>map.orientation</code>	a character vector with values "Pacific" and "Atlantic". It distinguishes Pacific-centered and Atlantic-centered maps. By default is "Pacific".
<code>radius</code>	deprecated argument

Author(s)

George Moroz <agricolamz@gmail.com>

Examples

```
map.feature(c("Kabardian", "Russian"))
```

oto_mangueanIC	<i>Oto-Manguean Inflectional Class Database Language identifiers</i>
----------------	--

Description

Language identifiers from Oto-Manguean Inflectional Class Database (<https://oto-manguean.surrey.ac.uk/>). 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 (<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](#)

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#) # `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	<i>Download PHOIBLE data</i>
-----------------	------------------------------

Description

This function downloads data from PHOIBLE (<https://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.

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	<i>Number of consonants and presence of ejectives</i>
-----------------------	---

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	<i>Get polygons from fixed distance circles around coordinates</i>
-------------------	--

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 polygons around points

polygon.points_kde	<i>Get kernel density estimation polygon from coordinates</i>
--------------------	---

Description

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

Usage

```
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	<i>Providers</i>
-----------	------------------

Description

List of all providers with their variations taken from leaflet package

Usage

```
providers
```

Format

A list of characters

Source

<https://github.com/leaflet-extras/leaflet-providers/blob/master/leaflet-providers.js>

sails.feature	<i>Download SAILS data</i>
---------------	----------------------------

Description

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

Usage

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

Arguments

<code>features</code>	A character vector that define with a feature ids from SAILS (e. g. "and1", "argex4-1-3").
<code>na.rm</code>	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](#), [phoible.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [uralex.feature](#), [valpal.feature](#), [wals.feature](#)

Examples

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

soundcomparisons

SOUNDCOMPARISONS's Language identifiers

Description

Language identifiers from SOUNDCOMPARISONS. This dataset is created for [soundcomparisons.feature](#) function.

Usage

```
soundcomparisons
```

Format

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

Details

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

LanguageName SOUNDCOMPARISONS language identifier

LanguageId Language Id

 soundcomparisons.feature

Download SOUNDCOMPARISONS data

Description

This function downloads data from SOUNDCOMPARISONS 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

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [eurasianphonology.feature](#), [eurasianphonology.feature](#)

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

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'))
```

uralex	<i>UraLex's Language identifiers</i>
--------	--------------------------------------

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:

uralex.name language name from database

glottocode Glottocodes

language language from lingtypology

uralex.feature	<i>Download UraLex data</i>
----------------	-----------------------------

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](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [valpal.feature](#), [vanuatu.feature](#), [wals.feature](#)

Examples

```
# uralex.feature()
```

url.lang	<i>Make a url-link to glottolog page for a language</i>
----------	---

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 <code>map.feature</code>

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', 'West Circassian', '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

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [vanuatu.feature](#), [wals.feature](#)

Examples

```
# valpal.feature()
```

vanuatu.feature	<i>Download Vanuatu Voices data</i>
-----------------	-------------------------------------

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

[abvd.feature](#), [afbo.feature](#), [autotyp.feature](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [wals.feature](#)

wals	<i>WALS's Language identifiers</i>
------	------------------------------------

Description

Language identifiers from WALS (<https://wals.info/>). This dataset is created for [wals.feature](#) function.

Usage

```
wals
```

Format

A data frame with 2678 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

<code>features</code>	A character vector that define with a feature ids from WALs (e. g. "1a", "21b").
<code>na.rm</code>	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](#), [bivaltyp.feature](#), [eurasianphonology.feature](#), [oto_mangueanIC.feature](#), [phoible.feature](#), [sails.feature](#), [soundcomparisons.feature](#), [uralex.feature](#), [valpal.feature](#), [vanuatu.feature](#)

Examples

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

Index

* datasets

- abvd, 3
- autotyp, 7
- bantu, 8
- circassian, 10
- countries, 11
- eurasianphonology, 12
- glottolog, 14
- iso_639, 20
- oto_mangueanIC, 31
- phoible, 32
- phonological_profiles, 33
- providers, 35
- soundcomparisons, 36
- uralex, 38
- wals, 41
- %>(imports), 17
- %>, 17
- abvd, 3
- abvd.feature, 3, 4, 5, 8–10, 13, 14, 17, 32, 33, 36, 37, 39–42
- afbo.feature, 4, 4, 8–10, 13, 14, 17, 32, 33, 36, 37, 39–42
- aff.lang, 5, 6, 12, 15, 16, 18–20, 22, 24, 25, 38, 40
- area.lang, 5, 6, 12, 15, 16, 18–20, 24, 25, 38, 40
- atlas.database, 6
- autotyp, 7
- autotyp.feature, 4, 5, 7, 8, 9, 10, 13, 17, 32, 33, 36, 37, 39–42
- bandwidth.nrd, 28, 29, 35
- bantu, 8
- bantu.feature, 8, 9
- bivaltyp.feature, 4, 5, 8, 10, 13, 14, 17, 32, 33, 36, 37, 39–42
- circassian, 10
- colorNumeric, 29
- countries, 11
- country.lang, 5, 6, 11, 16, 19, 22, 24, 25, 38, 40
- eurasianphonology, 12
- eurasianphonology.feature, 4, 5, 8, 10, 12, 13, 14, 17, 32, 33, 36, 37, 39–42
- frequency_list.feature, 13
- glottolog, 14
- gltc.iso, 15
- gltc.lang, 5, 6, 12, 16, 19, 22, 24, 25, 38, 40
- grambank.feature, 16
- imports, 17
- is.glottolog, 17
- iso.gltc, 18
- iso.lang, 5, 6, 12, 16, 19, 22, 24, 25, 38, 40
- iso3.iso1, 19
- iso_639, 20
- lang.aff, 21, 22, 23
- lang.country, 21
- lang.gltc, 22
- lang.iso, 21, 23
- lat.lang, 5, 6, 12, 15, 16, 18–20, 22, 23, 24, 25, 38, 40
- level.lang, 24
- long.lang, 5, 6, 12, 15, 16, 18–20, 22, 24, 25, 38, 40
- map.feature, 25
- oto_mangueanIC, 31
- oto_mangueanIC.feature, 4, 5, 8–10, 13, 14, 17, 31, 32, 33, 36, 37, 39–42
- phoible, 32

phoible.feature, [4](#), [5](#), [8–10](#), [13](#), [14](#), [17](#), [32](#),
[33](#), [36](#), [37](#), [39–42](#)
phonological_profiles, [33](#)
polygon.points_fd, [34](#)
polygon.points_kde, [34](#)
providers, [35](#)

sails.feature, [4](#), [5](#), [8–10](#), [13](#), [14](#), [17](#), [32](#), [33](#),
[35](#), [37](#), [39–42](#)
soundcomparisons, [36](#)
soundcomparisons.feature, [4](#), [5](#), [8](#), [10](#), [13](#),
[14](#), [17](#), [32](#), [33](#), [36](#), [37](#), [39–42](#)
subc.lang, [5](#), [6](#), [12](#), [16](#), [19](#), [22](#), [24](#), [25](#), [37](#), [40](#)

uralex, [38](#)
uralex.feature, [4](#), [5](#), [8–10](#), [13](#), [14](#), [17](#), [32](#),
[33](#), [36–38](#), [39](#), [40–42](#)
url.lang, [5](#), [6](#), [12](#), [16](#), [19](#), [22](#), [24](#), [25](#), [39](#)

valpal.feature, [4](#), [5](#), [8–10](#), [13](#), [14](#), [17](#), [32](#),
[33](#), [36](#), [37](#), [39](#), [40](#), [41](#), [42](#)
vanuatu.feature, [4](#), [5](#), [8](#), [10](#), [13](#), [14](#), [17](#), [32](#),
[33](#), [36](#), [37](#), [39](#), [40](#), [41](#), [42](#)

wals, [41](#)
wals.feature, [4](#), [5](#), [8](#), [10](#), [13](#), [14](#), [17](#), [32](#), [33](#),
[36](#), [37](#), [39–41](#), [42](#)