Package 'naijR'

November 6, 2026

Title Operations to Ease Data Analyses Specific to Nigeria

Version 0.7.1

Depends R (>= 3.6), grDevices, graphics, stats, utils

Imports RColorBrewer (>= 1.1.2), cli (>= 3.6.0), mapdata (>= 2.3.0), maps (>= 3.3.0), rlang (>= 0.4.0), sf (>= 1.0-12), stringi (>= 1.7.6),

Suggests ISOcodes, covr, here, knitr, lifecycle, purrr, readr, rmarkdown, testthat (>= 3.0.0), tibble, usethis

Description A set of convenience functions as well as geographical/political data about Nigeria, aimed at simplifying work with data and information that are specific to the country.

License GPL-3

LazyData TRUE

Encoding UTF-8

VignetteBuilder knitr

URL https://docs.ropensci.org/naijR/

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

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Victor Ordu [aut, cre] (ORCID: <https://orcid.org/0000-0003-3716-0668>), Laura DeCicco [ctb], Emily Riederer [rev], Margaret Siple [rev], Cagri Alican [rev]

Maintainer Victor Ordu <victorordu@outlook.com>

Repository CRAN

Date/Publication 2026-11-06 21:50:02 UTC

Contents

disambiguate_lga 2
fix_mobile 3
fix_region 4
lgas 5
lgas_nigeria 7
map_ng 7
ngdist 10
ng_distance 10
states 11
states_nigeria 13

Index **14**

`disambiguate_lga` *Disambiguate Synonymous States and LGAs*

Description

Some LGAs in Nigeria bear the name of the States to which they belong to. This function will apply an attribute to such an LGA to distinguish it from its State.

Usage

```
disambiguate_lga(lga, state = NULL, ...)
```

Arguments

`lga`	An object of class `lgas` of length `1L`.
`state`	The name of the State to which the LGA is to belong to.
`...`	Arguments to be passed to `menu`.

Details

For `state`, if it is not provided by the user, an interactive prompt will be presented to the user to select the appropriate state - but only in interactive sessions; if run as a batch command, this function will signal an error.

Value

The object of class `lgas` with the (possibly) modified `State` attribute.

Examples

```
obi.lga <- lgas("Obi")    # Warning
try(map_ng(obi.lga))      # Error

obi.benue <- disambiguate_lga(obi.lga, "Benue")

if (interactive())
  map_ng(obi.benue)
```

`fix_mobile` *Fix mobile numbers*

Description

Fixes up local mobile phone numbers to a uniform text format.

Usage

```
fix_mobile(x)
```

Arguments

`x`	Values to be fixed, represented either as `numeric` or `character` vectors.

Details

This format is specific to that used in a given location - for now the function is useful only for Nigeria mobile numbers, which come in the format expressed by the regex pattern `"^0[7-9][0-1][0-9]{8}$"`.

Value

The updated vector, usually the column of a data frame.

Note

There is an option for producing warnings on any mobile number entries that may have been removed from the vector, by setting the option `verbose` to `TRUE`.

Examples

```
fix_mobile("803-123-4567")    # Adds leading '0" and removes separators
```

`fix_region` *Fix Misspelling of Administrative Regions*

Description

Correct any misspelled names of administrative regions e.g. States and LGAs.

Usage

```
fix_region(x, ...)

## S3 method for class 'states'
fix_region(x, ...)

## S3 method for class 'lgas'
fix_region(x, interactive = FALSE, quietly = FALSE, graphic = FALSE, ...)

## Default S3 method:
fix_region(x, ...)

fix_region_manual(x, wrong, correct)
```

Arguments

`x`	An S3 object of class `states` or `lgas`. For `fix_region.default`, a character vector (or an object coercible to one) can be passed but only that for 'States' can be interpreted.
`...`	Arguments passed to methods.
`interactive`	Logical. When `TRUE`, the function prompts the user to interactively select the correct LGA names from a list of available options.
`quietly`	Logical; default argument is `FALSE`.
`graphic`	Whether to make use of native GUI elements (on Windows only).
`wrong`	The misspelled element(s) of `x`.
`correct`	The correction that is to be applied to the misspelled element(s)

Details

`fix_region` will look through a character vector and try to determine if State or LGA names have been wrongly entered. This presupposes that the atomic vector is of type `character`. It does not test any missing values in the vector, leaving them untouched.

`fix_region_manual` allows users to interactively and directly change update the spelling.

Value

The transformed object. If all names are correct, the object is returned unchanged.

Note

When passed a character vector of length 1L, in the case of a misspelled LGA, the function signals an error; the presumption is that a fix can readily be applied interactively. When all the items provided are misspelled, nothing happens, but the user is advised to use the appropriate constructor function so as to improve the accuracy of the repairs. When there is a mix of misspelled and properly spelled LGAs, other avenues for fixing the mistakes are available via mode `interactive`.

Examples

```
try(fix_region("Owerri north")) # ERROR
fix_region(c("Owerri north", "Owerri West"))

x <- c("Pankshen", "Pankshin", "Q'uan Pam")
is_lga(x)
x <- fix_region(x, quietly = TRUE)
is_lga(x)
fix_region_manual(x, "Q'uan Pam", "Qua'an Pan")
all(is_lga(x))
```

---

lgas *Objects for Representing the Local Government Areas (LGAs) of Nigeria*

Description

Objects for Representing the Local Government Areas (LGAs) of Nigeria

Usage

```
lgas(region = NA_character_, strict = FALSE, warn = TRUE)

is_lga(x)

as_lga(x)

## S3 method for class 'lgas'
print(x, ...)

## S3 method for class 'lgas'
c(...)

## S3 method for class 'lgas'
x[[i, exact = TRUE]]

## S3 method for class 'lgas'
x[i]
```

```
## S3 method for class 'lgas'
na.exclude(object, ...)
```

Arguments

region	Context-dependent. Either State(s) of the Federation or Local Government Area(s) - internal checks are performed to determine what applies. In cases where States are synonymous to LGAs, the default behaviour is to use the State as a basis for selecting the LGAs. This can be modified with `strict`. The default value is `NA_character_` and will return all 774 LGAs.
strict	logical; in the event of a name clash between State/LGA, return only the specified LGA when this argument is set to `TRUE`.
warn	logical; issue a warning when one or more elements are not actually Local Government Areas (or were misspelt).
x	An object of type `character`. This includes higher dimension object classes like `matrix` and `array`. For `as_lga`, a string representing a Local Government Area that shares its name with one of its States.
...	Arguments used for methods. See documentation of generic for details.
i, exact	See help file for `?Extract`
object	An object of class `regions`

Value

If length of `ng.state` == 1L, a character vector containing the names of Local Government Areas; otherwise a named list, whose elements are character vectors of the LGAs in each state. `is_lga` returms a vector the same length as the input object (each element that is not a valid Local Government Area will evaluate to `FALSE`); with `as_lga`, an object of class `lgas`.

Note

There are six (6) LGAs that share names with their State - Bauchi, Ebonyi, Gombe, Katsina, Kogi and Ekiti.

Examples

```
how_many_lgas <- function(state) {
  require(naijR)
  stopifnot(all(is_state(state)))
  cat(sprintf("No. of LGAs in %s State:", state),
    length(lgas(state)),
    fill = TRUE)
}
how_many_lgas("Sokoto")
how_many_lgas("Ekiti")
is_lga(c("Pankshen", "Pankshin"))

# With coercion
kt.st <- states("Katsina")  # Ensure this is a State, not an LGA.
kt.lg <- suppressWarnings(as_lga(kt.st))
```

```
is_state(kt.st)             # TRUE
is_lga(kt.lg)               # TRUE

## Where there's no ambiguity, it doesn't make sense to coerce
## This kind of operation ends with an error
## Not run:
as_state("Kano")
as_lga("Michika")

## End(Not run)
```

---

lgas_nigeria *Local Government Areas of Nigeria*

Description

A dataset containing the 774 Local Government Areas of Nigeria.

Usage

```
data(lgas_nigeria)
```

Format

A data frame with 774 rows and 2 columns

lga Local Government Area

state State of the Federation

gpz Geo-political zone

---

map_ng *Map of Nigeria*

Description

Maps of the Federal Republic of Nigeria that are based on the basic plotting idiom utilised by maps:map and its variants.

Usage

```
map_ng(
  region = character(),
  data = NULL,
  x = NULL,
  y = NULL,
  breaks = NULL,
  categories = NULL,
  excluded = NULL,
  exclude.fill = NULL,
  title = NULL,
  caption = NULL,
  show.neighbours = FALSE,
  show.text = FALSE,
  legend.text = NULL,
  leg.title,
  plot = TRUE,
  ...
)
```

Arguments

region	A character vector of regions to be displayed. This could be States or Local Government Areas.
data	An object containing data, principally the variables required to plot in a map.
x, y	Numeric object or factor (or coercible to one). See *Details*.
breaks	Numeric. A vector of length >= 1. If a single value i.e. scalar, it denotes the expected number of breaks. Internally, the function will attempt to compute appropriate category sizes or fail if out-of bounds. Where length is >= 3L, it is expected to be an arithmetic sequence that represents category bounds as for cut (applicable only to choropleth maps).
categories	The legend for the choropleth-plotted categories. If not defined, internally created labels are used.
excluded	Regions to be excluded from a choropleth map.
exclude.fill	Colour-shading to be used to indicate excluded regions. Must be a vector of the same length as excluded.
title, caption	An optional string for annotating the map.
show.neighbours	Logical; TRUE to display the immediate vicinity neighbouring regions/countries.
show.text	Logical. Whether to display the labels of regions.
legend.text	Logical (whether to show the legend) or character vector (actual strings for the legend). The latter will override whatever is provided by categories, giving the user additional control.
leg.title	String. The legend title. If missing, a default value is acquired from the data. To turn off the legend title, pass NULL.

plot	Logical. Turn actual plotting of the map off or on.
...	Further arguments passed to plot

Details

The default value for region is to print all State boundaries. data enables the extraction of data for plotting from an object of class data.frame. Columns containing regions (i.e. States as well as supported sub-national jurisdictions) are identified. The argument also provides context for quasiquotation when providing the x and y arguments.

For x and y, when both arguments are supplied, they are taken to be point coordinates, where x represent longitude and y latitude. If only x is supplied, it is assumed that the intention of the user is to make a choropleth map, and thus, numeric vector arguments are converted into factors i.e. number classes. Otherwise factors or any object that can be coerced to a factor should be used.

For plain plots, the col argument works the same as with map. For choropleth maps, the colour provided represents a (sequential) colour palette based on RColorBrewer::brewer.pal. The available colour options can be checked with getOption("choropleth.colours") and this can also be modified by the user.

If the default legend is unsatisfactory, it is recommended that the user sets the legend.text argument to FALSE; the next function call should be legend which will enable finer control over the legend.

Value

An object of class sf, which is a standard format containing the data used to draw the map and thus can be used by this and other popular R packages to visualize the spatial data.

Note

When adjusting the default colour choices for choropleth maps, it is advisable to use one of the sequential palettes. For a list of of available palettes, especially for more advanced use, review RColorBrewer::display.brewer.all.

See Also

vignette("nigeria-maps") for additional ways to use this function.

Examples

```
## Not run:
map_ng() # Draw a map with default settings
map_ng(states("sw"))
map_ng("Kano")
## End(Not run)
```

ngdist *Road Distances Between Nigerian State Capitals*

Description

A symmetric distance matrix containing approximate road (driving) distances in kilometres between all 37 Nigerian State capitals (including Abuja for the Federal Capital Territory).

Usage

```
data(ngdist)
```

Format

An object of class `dist` with 37 labels — one for each State capital. The labels are city names in title case (e.g. `"Lagos"`, `"Port Harcourt"`).

Details

The matrix holds $37 \times 36/2 = 666$ unique pairwise distances. Values are sourced from UNDP and represent typical road travel; actual distances may vary with route choice, road conditions, or detours.

For a convenient way to look up individual distances, see `ng_distance`.

See Also

`ng_distance` for querying a single pair of cities.

Examples

```
data(ngdist)
labels(ngdist)          # city names
as.matrix(ngdist)[1:5, 1:5]
```

ng_distance *Distance between two Nigerian state capitals*

Description

Retrieves the precomputed road (driving) distance between Nigerian state capitals. Both `a` and `b` can be vectors of the same length, in which case distances are computed element-wise.

Usage

```
ng_distance(a, b, unit = c("km", "miles"))
```

Arguments

a, b	Character vector. Name(s) of city/cities (case-insensitive). Must be the same length; vector recycling is not permitted.
unit	Character. Unit of distance to return. One of "km" (default) or "miles".

Details

Distances are approximate driving distances sourced from travel allowance guidance provided by UNDP Nigeria. They represent typical road travel and may vary due to route choice, road conditions, or detours.

Value

Numeric vector of road distances in the requested unit (rounded to 1 decimal place), with the same length as a and b.

Examples

```
ng_distance("Lagos", "Kano")
ng_distance("Abuja", "Port Harcourt", unit = "miles")
ng_distance("Ibadan", "enugu")  # case-insensitive
ng_distance(c("Lagos", "Abuja"), c("Kano", "Enugu"))  # vectorized
```

---

states *Objects for Representing the Federal States of Nigeria*

Description

Objects for Representing the Federal States of Nigeria

Usage

```
states(states, gpz = NULL, all = TRUE, warn = TRUE)

is_state(x)

as_state(x)

## S3 method for class 'states'
print(x, ...)

## S3 method for class 'states'
c(...)

## S3 method for class 'states'
x[[i, exact = TRUE]]
```

```
## S3 method for class 'states'
x[i]

## S3 method for class 'states'
na.exclude(object, ...)
```

Arguments

states	A character vector with strings representing one or more States of Nigeria. If missing, the function will return a states object containing all the States.
gpz	NULL (the default) or, case insensitively, one or more of the following strings: "nc", "ne", "nw", "se", "ss" and "sw" (see "Details").
all	logical; whether to include the Federal Capital Territory (FCT) in the result.
warn	logical; issue a warning when one or more elements are not actually States (i.e. they were misspelt).
x	For is_state a vector to be tested. For as_state, a string representing a State that shares its name with one of its Local Government Areas.
...	Arguments used for methods. See documentation of generic for details.
i, exact	See help file for ?Extract
object	An object of class regions

Details

gpz represents a geopolitical zone which, in the Nigerian context, is a national subdivision that groups contiguous states that bear certain socio-cultural and political similarities. Historically, they arise from sub-national administrative divisions known as 'Regions' that existed at the time of the country's independence. There are at present 6 such zones - North-Central, North-East, North-West, South-East,South-South and South-West.

For is_state, An element-wise check of a supplied vector is carried out. To test an entire vector and return a single boolean value, functions such as base::all or base::any should be used (see examples).

Value

The States of Nigeria as a whole or by zones, as an S3 object of class states. is_state returns a logical vector.of same length as the input. If the input object is not even of type character, return the object unaltered, with a warning. In the case of as_state, an object of class states.

Note

is_state throws a warning, when a missing value is among the elements. It works only for atomic vectors, throwing an error when this is not the case or when NULL is passed to it. This design decision was made to allow rapid iteration through data frames without interruption, since spelling mistakes tend to be common.

Examples

```
states()  # lists names of all States
states(gpz = "se")  # lists States in South-East zone
all(is_state(naijR::states()))
is_state(c("Maryland", "Baden-Baden", "Plateau", "Sussex"))

# With coercion
kt.st <- states("Katsina")  # Ensure this is a State, not an LGA.
kt.lg <- suppressWarnings(as_lga(kt.st))
is_state(kt.st)             # TRUE
is_lga(kt.lg)               # TRUE

## Where there's no ambiguity, it doesn't make sense to coerce
## This kind of operation ends with an error
## Not run:
as_state("Kano")
as_lga("Michika")

## End(Not run)
```

---

states_nigeria *States of Nigeria*

Description

A dataset of the 36 States and Federal Capital Territory of Nigeria.

Usage

```
data(states_nigeria)
```

Format

A `data.frame` with 37 rows and 2 columns

isocode ISO 3661 Alpha 2 code

state Name of the State or Territory

gpz Geo-political zone

Index

∗ **datasets**
    lgas_nigeria, 7
    ngdist, 10
    states_nigeria, 13
[.lgas *(*lgas*)*, 5
[.states *(*states*)*, 11
[[.lgas *(*lgas*)*, 5
[[.states *(*states*)*, 11

as_lga *(*lgas*)*, 5
as_state *(*states*)*, 11

c.lgas *(*lgas*)*, 5
c.states *(*states*)*, 11
cut, *8*

disambiguate_lga, 2

fix_mobile, 3
fix_region, 4
fix_region_manual *(*fix_region*)*, 4

is_lga *(*lgas*)*, 5
is_state *(*states*)*, 11

legend, *9*
lgas, 5
lgas_nigeria, 7

map, *9*
map_ng, 7
maps:map, *7*
menu, *2*

na.exclude.lgas *(*lgas*)*, 5
na.exclude.states *(*states*)*, 11
ng_distance, *10*, 10
ngdist, 10

plot, *9*
print.lgas *(*lgas*)*, 5
print.states *(*states*)*, 11

states, 11
states_nigeria, 13