A Quick Overview of the naijR Package

Talk with the Abuja R User Group

Victor Ordu
22 November, 2022

Outline

- Background
- Usage
- Prospects

BACKGROUND

Challenges

A suite of functions for:

- Data entry
- Data cleaning
- Accurate naming
- Visualization

Design Principles

- Open-source and open development
- Locally relevant
- Meet global standards
- Extensibility
- Simplicity

USAGE

Installation

- Stable version:

```
install.packages("naijR")
```

- Development version (dev branch)

```
# install.packages('remotes')
remotes::install_github("BroVic/naijR", ref = "dev")
```

Administrative Regions: States

- Using strings i.e. character vectors

```
1 s <- c("Adamawa", "Bauchi", "Borno", "Gombe", "Taraba", "Yobe")
2 s
```

```
[1] "Adamawa" "Bauchi"  "Borno"   "Gombe"   "Taraba"  "Yobe"
```

Administrative Regions: States

- Using `states` objects (S3 classes)

  A special vector constructed with the `states()` function:

```
1 library(naijR)
2 states()
```

```
Abia

Adamawa

Akwa Ibom

Anambra

Bauchi

Bayelsa

Benue

Borno
```

Administrative Regions: States

```
# Using earlier created vector
(stateobj <- states(s))
```

```
Adamawa

Bauchi

Borno

Gombe

Taraba

Yobe
```

Administrative Regions: States

Objects representing the sub-national divisions inherit from an abstract class `regions` to confer a particular behaviour.

- `regions` is an abstract class i.e. it does not have constructible objects, but exists to define shared behaviour between `states` and `lgas`.

```
1 class(stateobj)
```

```
[1] "states"    "regions"   "character"
```

Administrative Regions: States

`states` has additional arguments:

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

- `gpz` - a geopolitical zone (string)
- `all` - include FCT in the result? (logical)
- `warn` - notify if an element is not a valid State (logical)

Administrative Regions: Local Government Areas

- As with States, we can use character vectors with the names of the LGA.
- We can also create `lgas` objects - safer
- LGAs present an additional challenge:
  - Sheer number (774)
  - Duplication of LGAs
  - Ambiguity due to name-sharing

Because of this the function signature is more involved:

```
function (region = NA_character_, strict = FALSE, warn = TRUE)
NULL
```

Note:

- `region` - i.e. one or more States (character vector only) or selected LGAs.
- `strict` - use LGA when there is name-sharing (logical, default is `FALSE`).
- `warn` - notify of wrong spelling (logical).

Helper Functions

- `is_*` - are elements of the object what they claim to be?
- `fix_*` - carry out repairs.

Example:

```
1 nas <- "Nassarawa"
2 is_state(nas)
```

```
[1] FALSE
```

```
1 nas <- fix_region(nas)
```

```
Error: Incorrect region name(s); consider reconstructing 'x' with
`states()` or `lgas()` for a more reliable fix
```

```
1 nas <- fix_region(states(nas))
2 is_state(nas)
```

```
[1] TRUE
```

Fixing LGA spellings

```
am <-
  c("Amuwo-Olofin",
    "Amuwo-Odofin",
    "Amuwo-Odofin",
    "Amuwu-Odofin")

is_lga(am)
```

```
[1] FALSE  TRUE  TRUE FALSE
```

```
am |>
  fix_region() |>
  is_lga()
```

```
[1] TRUE TRUE TRUE TRUE
```

- Sometimes, LGAs cannot be repaired automatically
- This occurs when there are too name clashes
- The fixes can now be done interactively with the function `fix_region_manual()`.
- See the article that describes how this is done by running the following code:

```
vignette("interactive", "naijR")
```

Phone Numbers

Deal with poorly entered phone numbers and MS Excel mutilations using `fix_mobile`.

Phone Numbers

- Input numeric values…

```
1  fix_mobile(8034510441)
```

```
[1] "08034510441"
```

Phone Numbers

- or strings…

```
1  fix_mobile("8034510441")
```

```
[1] "08034510441"
```

Numbers that cannot be repaired are turned into missing values i.e. NAs.

```
nums <- c("8034510441", "070304", "09O14358956")
fix_mobile(nums)
```

```
[1] "08034510441" NA            "09014358956"
```

Note that one of the digits of `nums[3]` is not `0` but `O`. The function automatically repairs it.

Maps

- Plain plots - by default shows State boundaries

```
1  map_ng()
```

Maps

```
1  map_ng(lgas())
```

```
1 args(map_ng)
```

```
function (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.x = deprecated(), leg.y = deprecated(),
    leg.title, leg.orient = deprecated(), ...)
NULL
```

- Input options
  - A collection of States or LGAs
  - A data frame
  - A collection of coordinates

Combining Concepts

What do you expect to be the result of the following code?

```
map_ng("Bauchi")
```

Consider the following possibilities:

- Bauchi is the name of a State in Nigeria.
- Bauchi is the name of an LGA in Bauchi State of Nigeria.
- We could draw a map of:
  - Bauchi State
  - All the LGAs in Bauchi State
  - Bauchi LGA
- This informed the **polymorphism** used in the package.

```
map_ng("Bauchi")
```

```r
map_ng(states("Bauchi"), show.text = TRUE)
```

```
1  map_ng(lgas("Bauchi"), show.text = T)
```

```
1  map_ng(lgas("Bauchi", strict = T), show.text = T)
```

We can also create choropleth maps using the `map_ng()` function. For more info, read the vignette

```
vignette('nigeria-maps', 'naijR')
```

PROSPECTS

Some New Ideas

The package is not yet feature complete. Many changes still ahead:

- Provision of **richer** objects/methods
- Introduction of compiled code i.e. low-level constructs (C/C++)
- Link to Other Ecosystems
- A case for political wards
- More robust handling of phone numbers: Map to (inter)national standard
- Connection to geospatial packages

Collaboration

- The package is hosted publicly on GitHub and has a GPL-3 license, and thus open to modification, distribution, etc.
- How to contribute:
  - Issues
  - Pull_Requests
  - Documentation

Resources

- naijR website - **http://brovic.github.io/naijR**
- My personal blog - **https://victorordu.wordpress.com**

To contact me, visit my GitHub profile: **https://github.com/BroVic**

Error ×