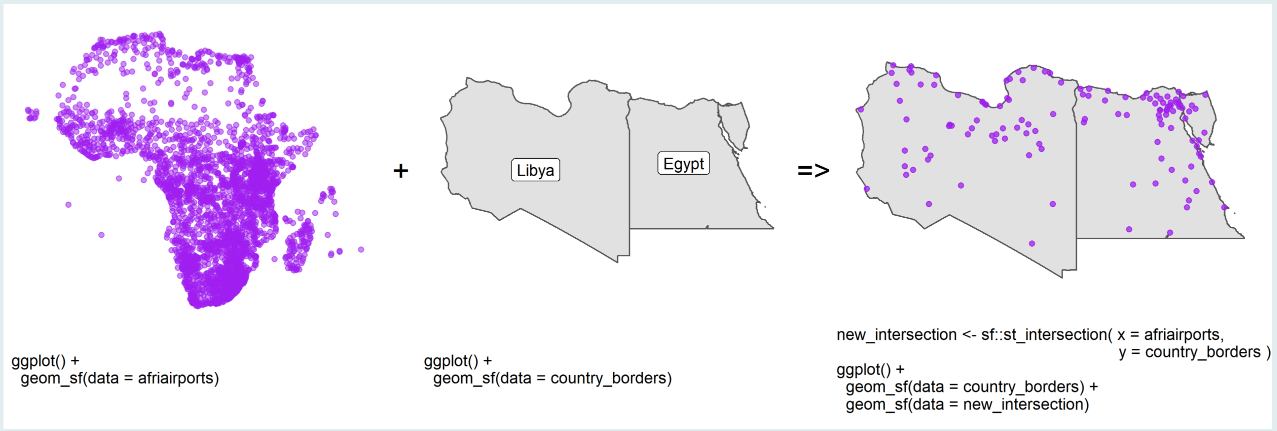
Boundary Data

Introduction

Country **borders** or **boundaries** can have several usages. For example, they can be used as background in Thematic maps or as delimiters of other Spatial data to ease the identification of spread patterns.

An example of the former is shown in Figure 1, where we retrieve the intersection between two spatial objects: points within polygons.



```
ggplot() +
  geom_sf(data = afriairports)
```

```
ggplot() +
  geom_sf(data = country_borders)
```

```
new_intersection <- sf::st_intersection( x = afriairports,
                                         y = country_borders )

ggplot() +
  geom_sf(data = country_borders) +
  geom_sf(data = new_intersection)
```

Figure 1. The st_intersection() function creates a new geometry with the shared portion of x and y.

However, the access to this type of data can have **different outputs**, for example, the *low* or *high* resolution of continent and country borders, or the *availability* of certain administrative levels. The choice of these outputs will depend of your needs!

In this lesson we are going to learn how to access continent, country and administrative level borders using `{rnaturalearth}`, `{rgeoboundaries}`, and `{geodata}` packages.

Learning objectives

1. Access to *low* resolution continent and country borders with {rnaturalearth}

2. Access to *high* resolution country and administrative level borders with {rgeoboundaries}

3. Access to *multiple* administrative level borders with {geodata}

Prerequisites

This lesson requires the following packages:

```
if(!require('pacman')) install.packages('pacman')

pacman::p_load(rnaturalearth,
               malariaAtlas,
               ggplot2,
               cholera,
               geodata,
               here,
               sf)

pacman::p_load_gh("afrimapr/afrilearndata",
                  "wmgeolab/rgeoboundaries")
```

Mapping country borders with {rnaturalearth}

As an introduction to plotting boundaries in R, let's look at how to draw a simple world map with country borders. The package {rnaturalearth} contains information to map all the countries in the world, among other things.

To obtain this map information, use the ne_countries() function, with the argument returnclass = "sf".

```
countries <- ne_countries(returnclass = "sf")
```

The code returns an sf object with the shapes for all countries.

Now, the countries object can be plotted very easily with the geom_sf() function of {ggplot2}:

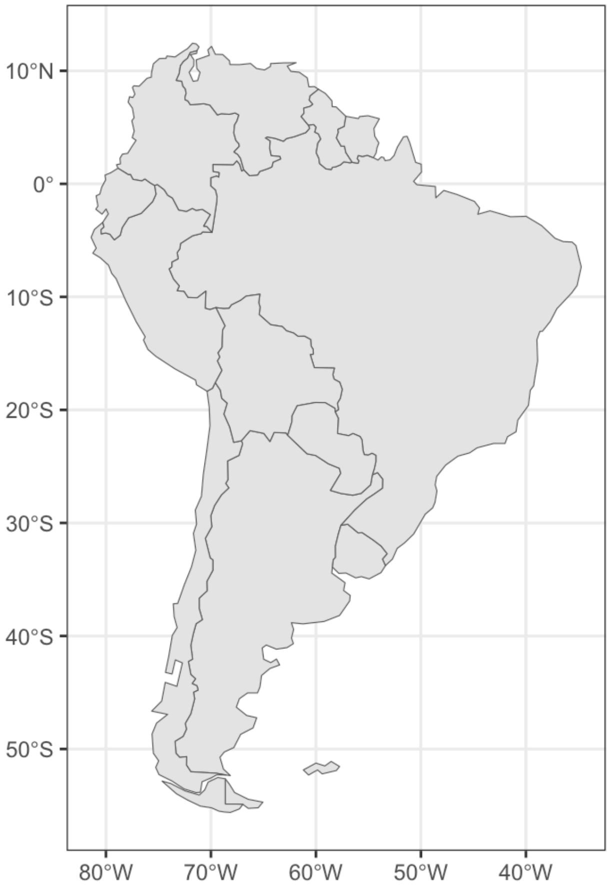
```
ggplot(data = countries) +
  geom_sf()
```

Wonderful! (Almost too easy!)

A single continent

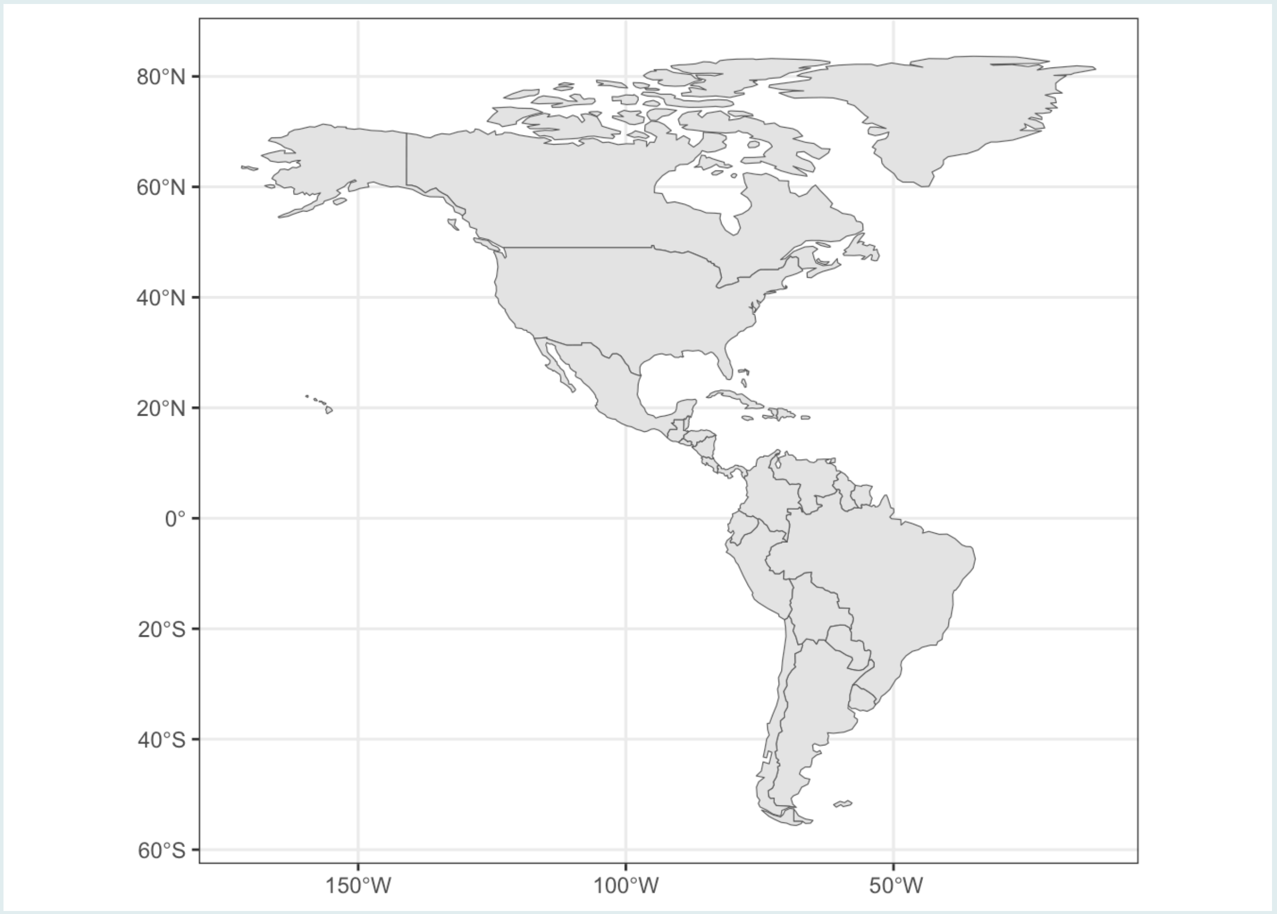
To subset to a **specific continent**, use the `continent` argument of `ne_countries()`:

```r
# Countries in South America
south_am <- ne_countries(returnclass = "sf",
                         continent = "south america") # 👈👈👈👈

ggplot(data = south_am) +
  geom_sf()
```



The `continent` argument can accept a vector with multiple continents:

```
# Countries in north and south america
north_south_am <-
  ne_countries(returnclass = "sf",
               continent = c("north america", "south america"))

ggplot(data = north_south_am) +
  geom_sf()
```



◘ Use `ne_countries()`, `ggplot()` and `geom_sf()` to plot a single map of all the countries in the Asia and Africa continent

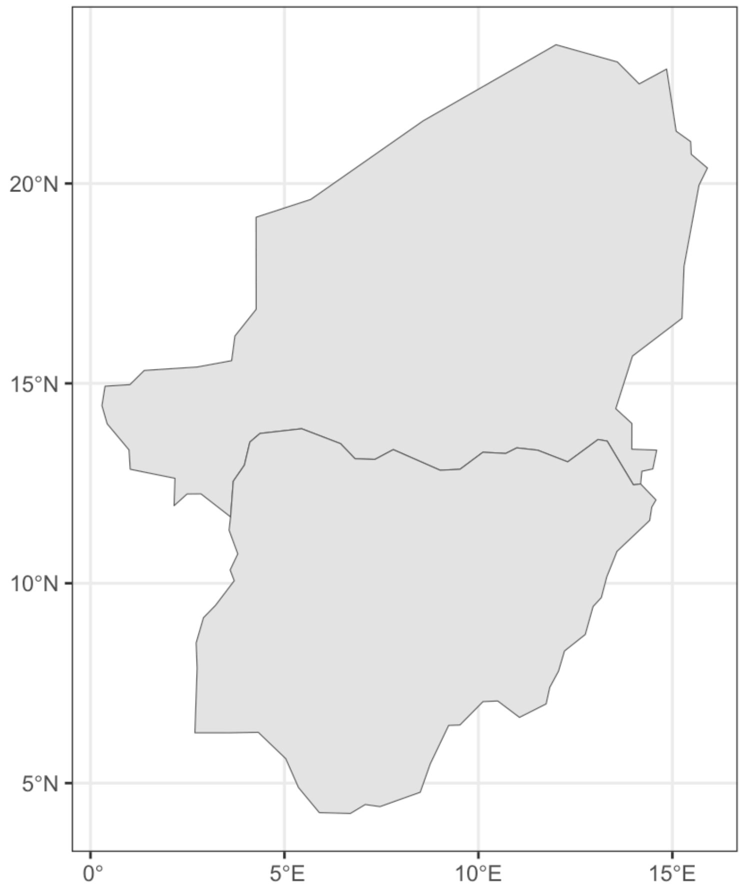
RSTUDIO CLOUD

```
asia_africa <-
  .........(returnclass = "sf",
            ......... = c(".........", "........."))

ggplot(data = asia_africa) +
  geom_sf()
```

Multiple countries

To subset to a **specific country** or specific countries, use the `country` argument:

```
# Map of Nigeria and Niger
nigeria_niger <- ne_countries(returnclass = "sf",
                              country = c("nigeria", "niger"))

ggplot(data = nigeria_niger) +
  geom_sf()
```



RSTUDIO CLOUD

▪ Use `ne_countries()`, `ggplot()` and `geom_sf()` to plot a single map of the national borders of China and Indonesia

```
china_indonesia <-
  .........(returnclass = "sf",
            ......... = c(".........", "........."))

ggplot(data = china_indonesia) +
  geom_sf()
```

Mapping country borders with `{rgeoboundaries}`

`{rnaturalearth}` is useful for accessing continents and country borders that do not need much boundary resolution. `{rgeoboundaries}` is an alternative package that provides access to high resolution country boundaries.

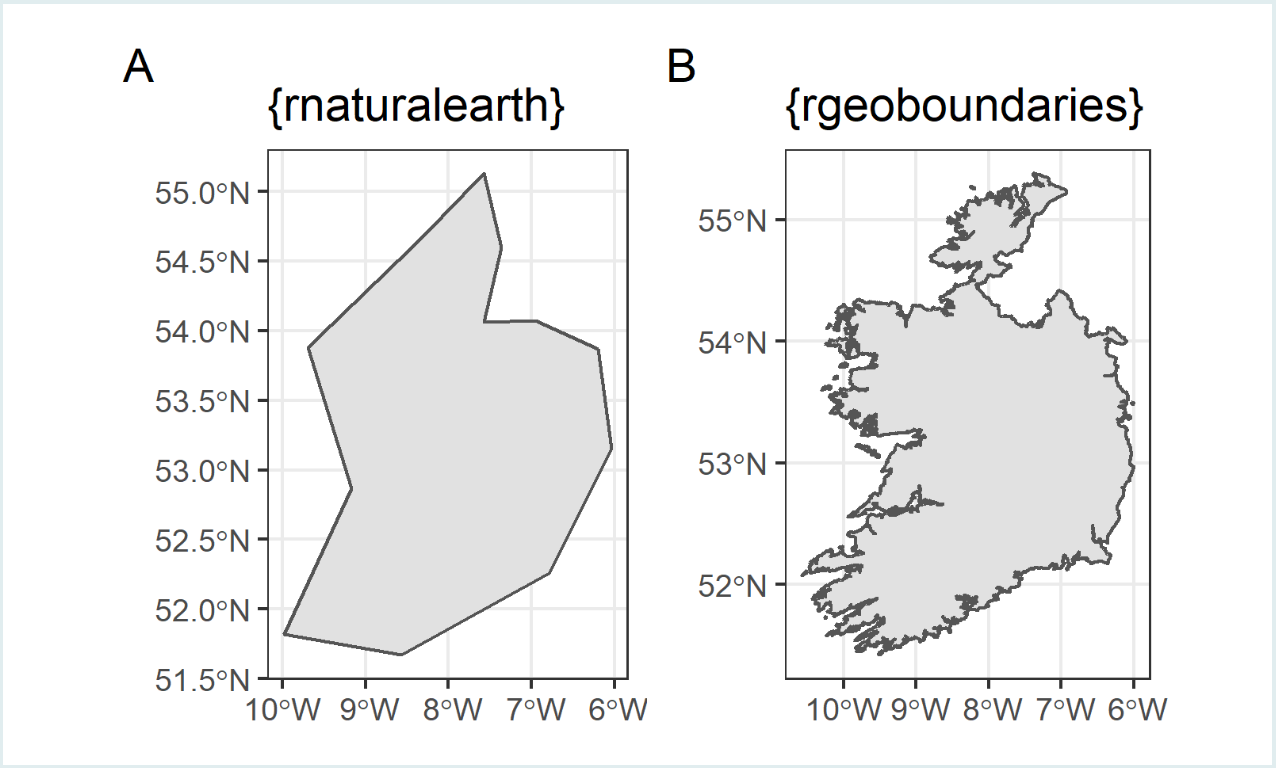


Figure 2. Ireland according to {rnaturalearth} and {rgeoboundaries} packages.

The `{rgeoboundaries}` package is a client for the geoBoundaries API, providing country political administrative boundaries.

A single country

To download boundaries of countries we use the `geoboundaries()` function of `{rgeoboundaries}`. For example, we can download the administrative boundary of Zimbabwe and assign it to a variable called `zimbabwe_boundary` as follows.

```
zimbabwe_boundary <- geoboundaries(country = "Zimbabwe")
```

KEY POINT

- The `zimbabwe_boundary` is a `"sf"` class object.
- `{ggplot2}` allows us to easily visualise **simple feature** objects using the `geom_sf()` function.
- It can be used to plot the administrative boundary of Zimbabwe as follows:

```
ggplot(data = zimbabwe_boundary) +
  geom_sf()
```

PRACTICE

(in RMD)

Download the boundaries of `Sierra Leone` using the `geoboundaries()` function.

```
q1 <- ________(________ = "Sierra Leone")
q1
```

Different administrative levels

If available, lower levels of administrative boundaries in countries can be downloaded too. We just have to pass the administrative level as an argument in the `geoboundaries()` function.

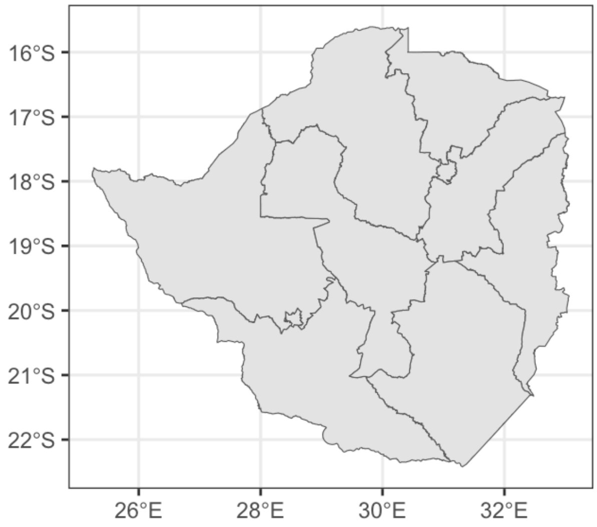
Administrative **level 1** (1) is the highest level, while administrative **level 5** (5) is the lowest. This means the country will be further sub-divided into administrative divisions as the Administrative level progresses from 1 to 5.

See how the first and second administrative level boundaries of Zimbabwe are downloaded below.

```
# downloading administrative level 1 boundaries
zimbabwe_boundaries_adm1 <- geoboundaries(country = "Zimbabwe",
                                          adm_lvl = 1)
```

```
## [1] "WARNING: geoBoundaries now provides two only types of boundaries:
simplified and unsimplified.All other types are deprecated. If you selected
SSCGS or SSCU it will be changed to simlified, HPSCU will be changed to
usimplifed "
```

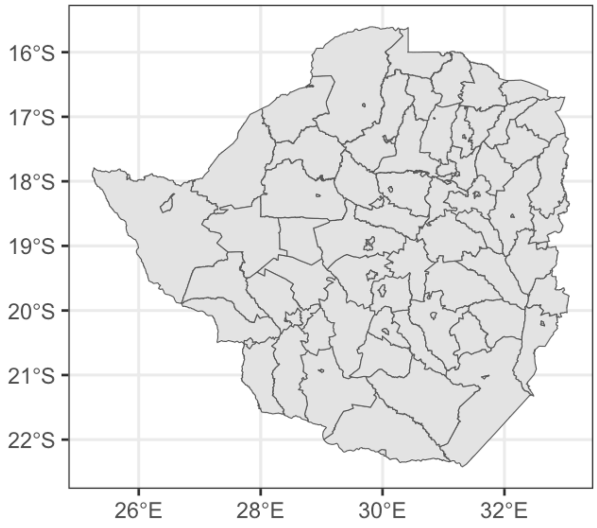
```
ggplot(data = zimbabwe_boundaries_adm1) +
  geom_sf()
```



```r
# downloading administrative level 2 boundaries
zimbabwe_boundaries_adm2 <- geoboundaries(country = "Zimbabwe",
                                         adm_lvl = 2)
```

```
## [1] "WARNING: geoBoundaries now provides two only types of boundaries:
simplified and unsimplified.All other types are deprecated. If you selected
SSCGS or SSCU it will be changed to simlified, HPSCU will be changed to
usimplifed "
```

```r
ggplot(data = zimbabwe_boundaries_adm2) +
  geom_sf()
```



PRACTICE

(in RMD)

Download the third administrative level boundaries of Sierra Leone, using the geoboundaries() function.

```
q2 <- geoboundaries(country = "Sierra Leone", ________ =
________)
q2
```

PRO TIP

We can also download the boundaries of **multiple countries** together by including the names of countries as a vector class object like: c("country_01","country_02").

See how the second administrative level boundaries of adjacent countries like Zimbabwe and Mozambique are downloaded and plotted below.

```
zimbabwe_mozambique_adm2 <-
  geoboundaries(country = c("Zimbabwe", "Mozambique"),
                adm_lvl = 2)
```

```
ggplot(data = zimbabwe_mozambique_adm2) +
  geom_sf()
```

Mapping country borders with {geodata}

A limitation of {rgeoboundaries} is that it only provides one column with the name of all the borders in that level:

```
geoboundaries(country = "Bolivia", adm_lvl = 3) %>%

  as_tibble() %>%
  select(contains("name"))
```

```
## [1] "WARNING: geoBoundaries now provides two only types of boundaries:
simplified and unsimplified.All other types are deprecated. If you selected
SSCGS or SSCU it will be changed to simlified, HPSCU will be changed to
usimplifed "
```

```
## # A tibble: 339 × 1
##    shapeName
##    <chr>
##  1 Esmeralda
##  2 Santuario de Quillacas
##  3 Pampa Aullagas
##  4 Llallagua
##  5 Caripuyo
##  6 Aiquile
##  7 Sacaca
##  8 Acasio
##  9 Huanuni
## 10 Huachacalla
## # ℹ 329 more rows
```

The gadm() function from the {geodata} additionally provides the name of all the levels above the one requested:

```
gadm(country = "Bolivia", level = 3, path = tempdir()) %>%

  as_tibble() %>%
  select(starts_with("NAME_"))
```

```
## # A tibble: 344 × 3
##    NAME_1 NAME_2                 NAME_3
##    <chr>  <chr>                  <chr>
##  1 Beni   Cercado                San Javier
##  2 Beni   Cercado                Trinidad
##  3 Beni   General José Ballivián Puerto Menor de Rurrenabaq…
##  4 Beni   General José Ballivián Reyes
##  5 Beni   General José Ballivián San Borja
##  6 Beni   General José Ballivián Santa Rosa
##  7 Beni   Iténez                 Baures
##  8 Beni   Iténez                 Huacaraje
##  9 Beni   Iténez                 Magdalena
## 10 Beni   Mamoré                 Puerto Siles
## # ℹ 334 more rows
```

RSTUDIO CLOUD

Use gadm() to download the third administrative level boundaries of Sierra Leone.

```
q_geodata <- ________(country = "Sierra Leone",
                      _______ = ________,
                      path = tempdir())
q_geodata
```

This *multiple column output* is useful for situations where you want to `filter()` the subdivisions (level 3) from a specific region (level 1) of departments (level 2).

For example, let's filter the *municipalities* (level 3) from the *department* (level 1) called `Santa Cruz`, in Bolivia:

```
gadm(country = "Bolivia", level = 3, path = tempdir()) %>%

  as_tibble() %>%
  select(starts_with("NAME_")) %>%

  filter(NAME_1 == "Santa Cruz") #👈👈👈👈👈👈👈👈👈👈👈👈👈👈
```

```
## # A tibble: 56 × 3
##    NAME_1     NAME_2         NAME_3
##    <chr>      <chr>          <chr>
##  1 Santa Cruz Andrés Ibáñez  Cotoca
##  2 Santa Cruz Andrés Ibáñez  El Torno
##  3 Santa Cruz Andrés Ibáñez  La Guardia
##  4 Santa Cruz Andrés Ibáñez  Porongo
##  5 Santa Cruz Andrés Ibáñez  Santa Cruz de la Sierra
##  6 Santa Cruz Angel Sandoval San Matías
##  7 Santa Cruz Chiquitos      Pailón
##  8 Santa Cruz Chiquitos      Roboré
##  9 Santa Cruz Chiquitos      San José
## 10 Santa Cruz Cordillera     Boyuibe
## # ℹ 46 more rows
```

But, how can we make a province map with *this* output?

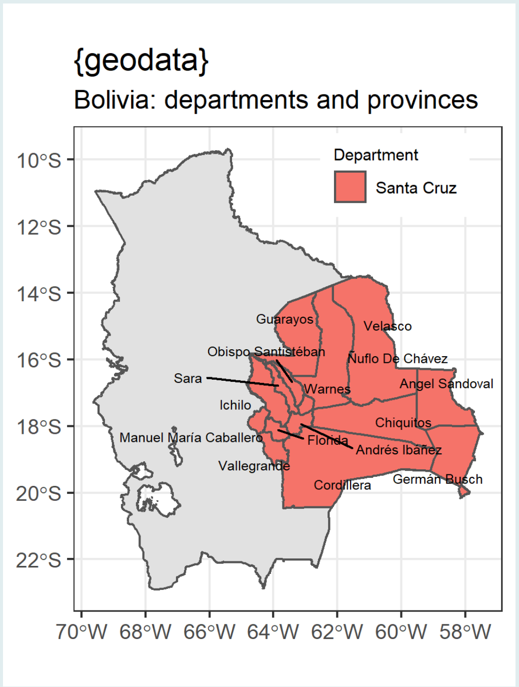


Figure 3. Province map from the Santa Cruz department in Bolivia.

The header of *this* object details that it is of class `SpatVector`. **It is not an `sf` object!**

```
gadm(country = "Bolivia", level = 3, path = tempdir())
```

```
👉 class       : SpatVector 👈
   geometry    : polygons
   dimensions  : 344, 16  (geometries, attributes)
   extent      : -69.64525, -57.45443, -22.90657, -9.670923
   coord. ref. : +proj=longlat +datum=WGS84 +no_defs
```

Wrap up

In this lesson, we have learned how to **access** *low* and *high* resolution continent, country and multiple administrative level borders using `{rnaturalearth}`, `{rgeoboundaries}`, and `{geodata}` packages.

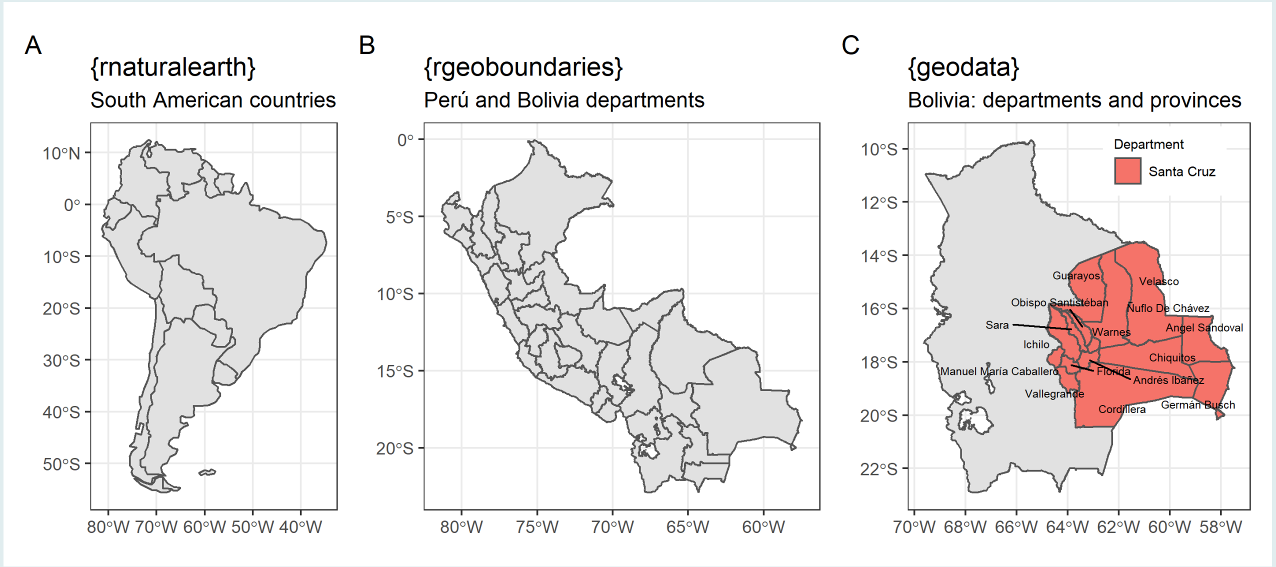


Figure 4. Advantages for each R package in different spatial scales. (A) Low resolution continent boundaries, (B) High resolution boundaries of multiple countries and one administrative level at the time, (C) Multiple administrative level boundaries in the same output.

Now, we need to learn how to **convert** foreign spatial objects to `sf`, to keep making `ggplot2` maps!

Contributors

The following team members contributed to this lesson:

ANDREE VALLE CAMPOS

R Developer and Instructor, the GRAPH Network
Motivated by reproducible science and education

LAURE VANCAUWENBERGHE

Data analyst, the GRAPH Network
A firm believer in science for good, striving to ally programming, health and education

KENE DAVID NWOSU

Data analyst, the GRAPH Network
Passionate about world improvement

References

Some material in this lesson was adapted from the following sources:

- *Seimon, Dilinie. Administrative Boundaries.* (2021). Retrieved 15 April 2022, from https://rspatialdata.github.io/admin_boundaries.html
- *Varsha Ujjinni Vijay Kumar. Malaria.* (2021). Retrieved 15 April 2022, from https://rspatialdata.github.io/malaria.html
- *Batra, Neale, et al. The Epidemiologist R Handbook. Chapter 28: GIS Basics*. (2021). Retrieved 01 April 2022, from https://epirhandbook.com/en/gis-basics.html
- *Lovelace, R., Nowosad, J., & Muenchow, J. Geocomputation with R. Chapter 2: Geographic data in R*. (2019). Retrieved 01 April 2022, from https://geocompr.robinlovelace.net/spatial-class.html
- *Moraga, Paula. Geospatial Health Data: Modeling and Visualization with R-INLA and Shiny. Chapter 2: Spatial data and R packages for mapping*. (2019). Retrieved 01 April 2022, from https://www.paulamoraga.com/book-geospatial/sec-spatialdataandCRS.html