Geocoding with R

The SF package

Andrew Wells & Fernanda Candido Gomes
Hertie School | I2DS Tools for Data Science workshop

Table of contents

What is the package good for?

Geocoding, spatial analysis and SF

What is **geocoding**? 🌍 🔍

Process of *taking an address or a name of a place and turning it into a geographic position on the earth's surface.*

☑ Input Data: Relative or Absolute

🔍 Coordinate System: Longitude and Latitude

The performance of analytic tasks that explicitly incorporate the spatial properties of a dataset.

Relevance for Public Policies

- public health, such as the pandemic evolution
- security, such as crime trends
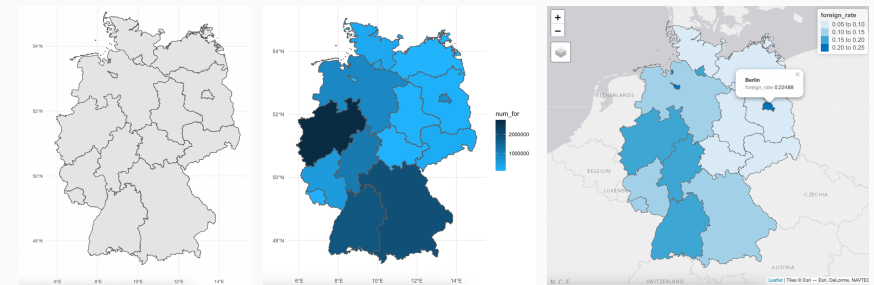
Adding more information conveys and increasingly rich and informative story

✌️ In its best form, allows people to easily understand complex geographical information.✌️

The **SF** package 🖥️ 🖥️ 🖥️

- Geographic information science has been performed in a geographic information system ("GIS"), which is an integrated software platform for the management, processing, analysis, and visualization of geographic data

- R packages exist for handling these tasks, allowing R to function as a capable substitute
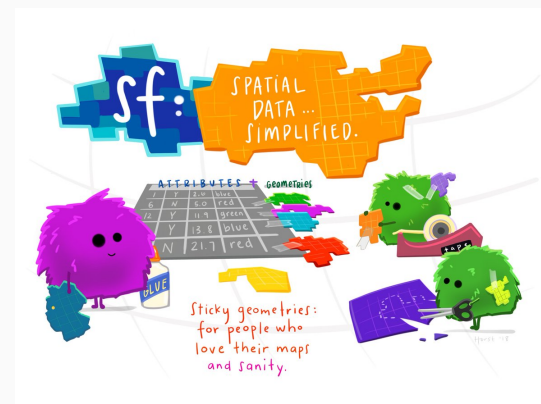
- The next-generation alternative to sp for spatial data analysis in R
- Advantages:
  - sf objects can be treated as data frames in most operations
  - sf function names are relatively consistent and intuitive ("st_")
  - sf functions can be combined using %>% operator
  - 👍 integrates seamlessly with **tidyverse tools** 👍

How can we use it?

Geometry basics

Basics

Simple features

thing or a object, which have "a **geometry** describing where on Earth is located (*spatial attribute*), and other attributes, which describe other properties (non-spatial attributes)"

- Coordinates: X and Y (longitude and latitude), Z (altitude) and M (denotes some associated measure, such as time of measurement)
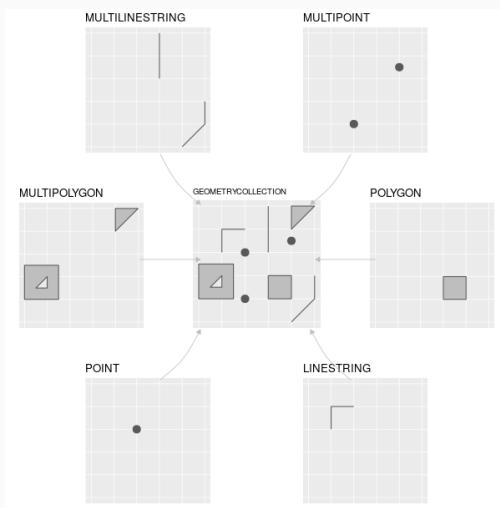

- Geometry types

type	description
POINT	zero-dimensional geometry containing a single point
LINESTRING	sequence of points connected by straight, non-self intersecting line pieces; one-dimensional geometry
POLYGON	geometry with a positive area (two-dimensional); sequence of points form a closed, non-self intersecting ring; the first ring denotes the exterior ring, zero or more subsequent rings denote holes in this exterior ring
MULTIPOINT	set of points; a MULTIPOINT is simple if no two Points in the MULTIPOINT are equal
MULTILINESTRING	set of linestrings
MULTIPOLYGON	set of polygons
GEOMETRYCOLLECTION	set of geometries of any type except GEOMETRYCOLLECTION

Shape files

- The way as *geographic information* is normally shared
- zip file with a **.shp**, which stores the geographic coordinates of the geographic features (e.g. country, state, county)

Key Features

Tools

SF: reading, writing, handling and manipulating

st_read	st_as_sf
imports a spatial data file and converts it to a simple feature *data frame*	*convert* foreign object to an sf object

read_sf is an alias for st_read, with some modified default arguments

Pratical Application

Tutorial Preview

How to?

Downloading and Visualizing geometric data

```r
R> # Save file as a .zip with link
R> brazil_file ← "https://biogeo.ucdavis.edu/data/diva/adm/BRA_adm.zip"
R>
R> # Download .zip
R> download.file(brazil_file, destfile = "BRA.zip")
R>
R> # Unzip the file
R> unzip("BRA.zip")
R>
R> # Examine the file .zip file (list()) to find the shapefile and then save the shapefile as a dataframe
R> brazil ← sf::st_read("BRA_adm1.shp")
```

The Geometry column

```r
R> brazil %>%
+    dplyr::select(geometry) %>%
+    knitr::kable(col.names = c("Geometry")) %>%
+    kableExtra::kable_minimal()
```

Geometry
MULTIPOLYGON (((-73.33251 -...
MULTIPOLYGON (((-35.90153 -...
MULTIPOLYGON (((-50.02403 0...
MULTIPOLYGON (((-67.32623 2...
MULTIPOLYGON (((-38.69708 -...
MULTIPOLYGON (((-38.47542 -...
MULTIPOLYGON (((-48.03603 -...
MULTIPOLYGON (((-40.88403 -...
MULTIPOLYGON (((-50.15817 -...

But if I don't have the geometry column?

Longitude and Latitude

```r
R> coordinates_br ← readr::read_csv("br.csv")
```

You will need to check the **CRS** (Coordinate Reference System) code

```r
R> sf::st_crs(brazil)
```

In our case: **"EPSG", 4326**

Then, conversion

```r
R> coord_geo ← coordinates_br %>%
+    sf::st_as_sf(coords = c("lng", "lat"), crs = 4326)
R>
R> coord_geo %>%
+    dplyr::select(geometry) %>%
+    knitr::kable(col.names = c("Geometry")) %>%
+    kableExtra::kable_minimal()
```

Geometry
POINT (-46.6339 -23.5504)
POINT (-43.1964 -22.9083)
POINT (-43.9419 -19.9281)
POINT (-47.8828 -15.7939)
POINT (-38.5108 -12.9708)

Next Steps

What about when I don't have Longitude and Latitude?

🌐 🔑 **geocoding**

And what about spatial analysis?

```
R> brplot ← ggplot2::ggplot() +
+   geom_sf(data = brazil) +
+   geom_sf(data = coord_geo %>%
+             dplyr::filter(!is.na(capital)), color = '
+   geom_sf_label(data = coord_geo %>%
+                 dplyr::filter(population_proper ≥ 1⁵
+                 aes(label = city),  size = 3, hjust =
```
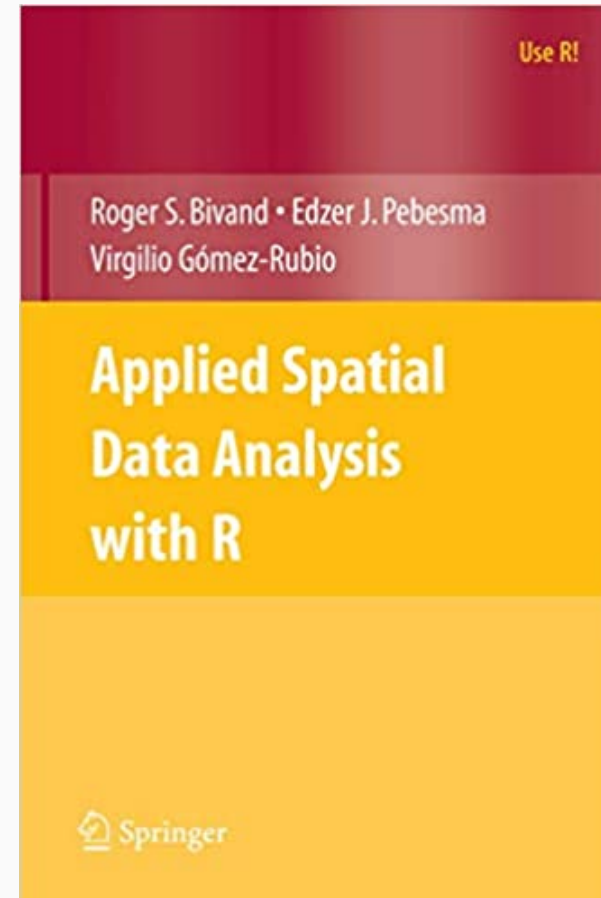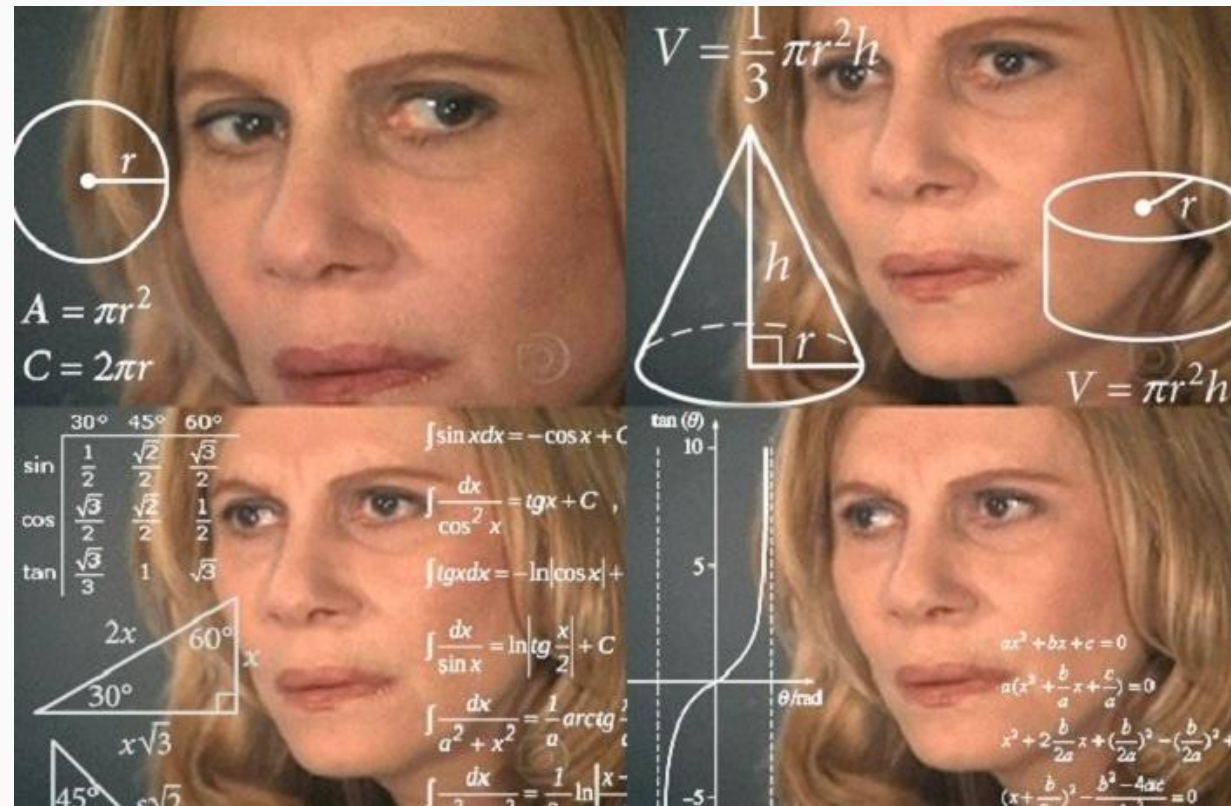
```
R> brplot
```

Learn More!

Where go to learn more? 🤓 📖

- Geocoding

- Geocoding with R

- How to use the sf package in R

- GIS in R with sf

- Geocomputation with R

- Simple Features for R

- sf Package

ME TRYING TO

FIND LONGITUDE AND LATITUDE