

Geocoding with R

The SF package

Andrew Wells & Fernanda Candido Gomes

Hertie School | [I2DS Tools for Data Science workshop](#)

Table of contents

1. What is the package good for?
2. How can we use it?
3. Key Features
4. Practical Application
5. Learn More

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

Why do spatial analysis?

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

Visibility and votes: A spatial analysis of anti-immigrant voting in Sweden

Sarah Valdez*

*Carlos III–Juan March Institute, Universidad Carlos III, C/ Madrid, 135 Edificio 18, Oficina 18.02F07, 29803 Getafe, Madrid, Spain. Email: svaldez@marichus

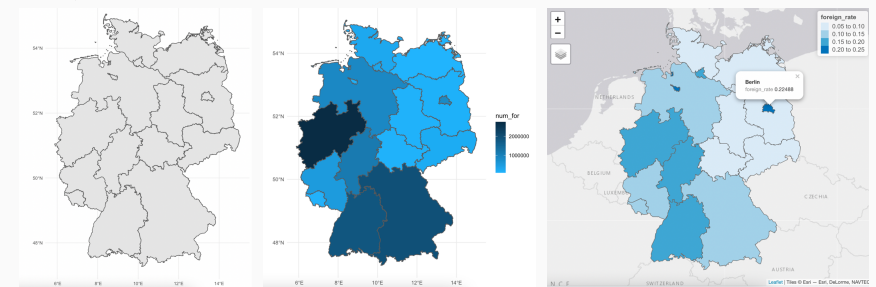
Abstract

The mechanisms by which negative attitudes toward immigrants become votes for anti-immigrant parties are not fully understood. Yet, voting for political parties with anti-immigrant platforms is arguably the most common expression of these sentiments in Europe. I use anti-immigrant attitudes as a starting point and hypothesize that superficial intergroup contact, or immigrant 'visibility', brings these attitudes to the fore as politically salient. A spatial analysis of electoral data from each polling station in Sweden for the 2010 parliamentary election ($n=5,688$) provides support for the hypothesis. Much of the variance in district-level voting can be accounted for by the percent of non-western residents in adjacent neighborhoods. The findings suggest that the probability of anti-immigrant attitudes translating into votes increases in neighborhoods where residents are likely to have fleeting contact with immigrants and I test this further with a city-level case study. I collected observational data on the visibility of non-westerners in a mid-size Swedish city and find that votes for the Sweden Democrats are above the national average where immigrants are most visible. Furthermore, the effect of non-western residents on anti-immigrant voting is most pronounced in regions without histories of significant non-western immigration, suggesting that the negative effects of superficial contact diminish over time.

Keywords: immigration, voting, contact, attitudes

Relevance for Public Policies

- public health, such as the pandemic evolution
- security, such as crime trends

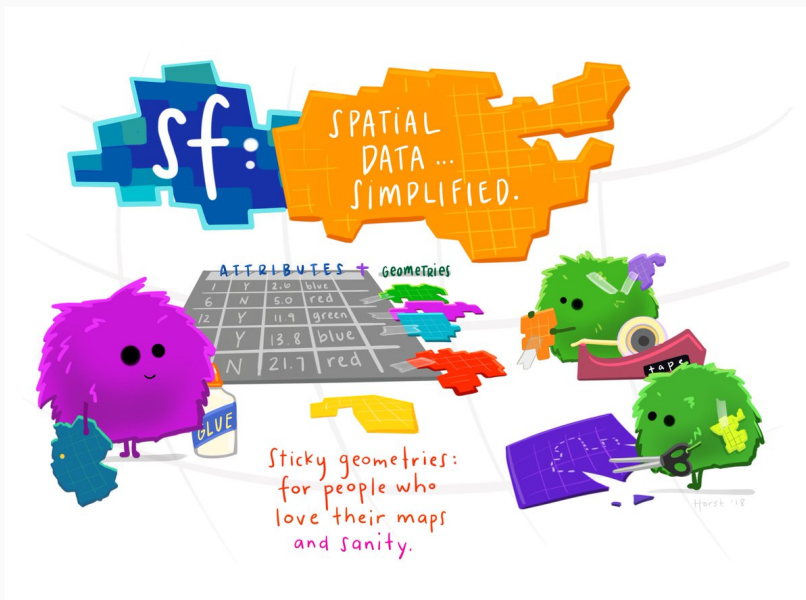
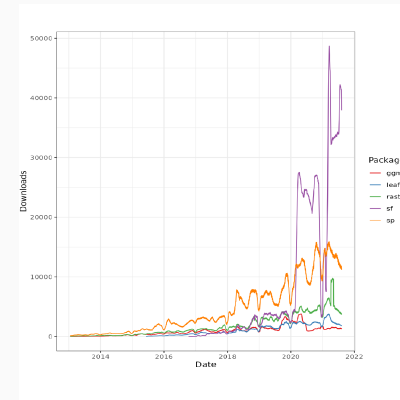


In its best form, allows people to easily understand complex geographical information: *Adding more information conveys and increasingly rich and informative story*

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
 - sf functions can be combined using %>% operator
 - 📄 integrates seamlessly with **tidyverse tools** 📄 / 17

How can we use it??

Geometry 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)"

- Dimensions or 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 simple

st_read

imports a spatial data file and converts
it to a simple feature *data frame*

st_as_sf

convert foreign object to an sf object

Spatial manipulation with sf: : CHEAT SHEET

The sf package provides a set of tools for working with geospatial vectors, i.e. points, lines, polygons, etc.



Geometric confirmation

- `st_contains(x, y, ...)` Identifies if x is within y (i.e. point within polygon)
- `st_covered_by(x, y, ...)` Identifies if x is completely within y (i.e. polygon completely within polygon)
- `st_covers(x, y, ...)` Identifies if any point from x is outside of y (i.e. polygon outside polygon)
- `st_crosses(x, y, ...)` Identifies if any geometry of x have commonalities with y
- `st_disjoint(x, y, ...)` Identifies when geometries from x do not share space with y
- `st_equals(x, y, ...)` Identifies if x and y share the same geometry
- `st_intersects(x, y, ...)` Identifies if x and y geometry share any space
- `st_overlaps(x, y, ...)` Identifies if geometries of x and y share space, are of the same dimension, but are not completely contained by each other
- `st_touches(x, y, ...)` Identifies if geometries of x and y share a common point but their interiors do not intersect
- `st_within(x, y, ...)` Identifies if x is in a specified distance to y



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

Geometric operations

- `st_boundary(x)` Creates a polygon that encompasses the full extent of the geometry
- `st_buffer(x, dist, nQuads=512)` Creates a polygon covering all points of the geometry within a given distance
- `st_centroid(x, ...)` of_largest_polygon Creates a point at the geometric centre of the geometry
- `st_convex_hull(x)` Creates geometry that represents the minimum convex geometry of x
- `st_line_merge(x)` Creates linestring geometry from sewing multi (linestring) geometry together
- `st_node(x)` Creates nodes on overlapping geometry where nodes do not exist
- `st_point_on_surface(x)` Creates a point that is guaranteed to fall on the surface of the geometry
- `st_polygonize(x)` Creates polygon geometry from linestring geometry
- `st_segmentize(x, distMaxLength, ...)` Creates linestring geometry from x based on a specified length
- `st_simplify(x, preserveTopology, tolerance)` Creates a simplified version of the geometry based on a specified tolerance



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

Geometry creation

- `st_triangulate(x, tolerance, bOnlyEdges)` Creates polygon geometry as triangles from point geometry
- `st_voronoi(x, envelope, tolerance, bOnlyEdges)` Creates polygon geometry covering the envelope of x, with x at the centre of the geometry
- `st_point(x, c(numeric vector), dim = "XYZ")` Creating point geometry from numeric values
- `st_multipoint(x = matrix(numeric values in rows), dim = "XYZ")` Creating multi point geometry from numeric values
- `st_linestring(x = matrix(numeric values in rows), dim = "XYZ")` Creating linestring geometry from numeric values
- `st_multilinestring(x = list(numeric matrices in rows), dim = "XYZ")` Creating multi linestring geometry from numeric values
- `st_polygon(x = list(numeric matrices in rows), dim = "XYZ")` Creating polygon geometry from numeric values
- `st_multipartpolygon(x = list(numeric matrices in rows), dim = "XYZ")` Creating multi polygon geometry from numeric values



`ggplot() +
geom_sf(data = st_intersects(schec, st_buffer(schec, 1000))`

This cheatsheet presents the sf package (Edzer Pebesma 2018) in version 0.8-3. See <https://github.com/r-spatial/sf> for more details.

CC-BY Ryan Garnett <http://github.com/ryangarnett>
<https://creativecommons.org/licenses/by/4.0/>

Practical Application

Tutorial Preview

How to?

Downloading and Visualizing geometric data

```
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::read_sf("BRA_adm1.shp")
```

The Geometry column

```
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 -...

But if I don't have a geometry column?

- Longitude and Latitude

```
R> coordinates_br <- readr::read_csv("br.csv")
```

You will need to check the **CRS** (coordinate reference system) code

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

In our case: "EPSG",4326

Then, conversion

```
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)

Next Steps

What about when I don't have longitude and latitude?



And what about spatial analysis?

```
R> brplot <- ggplot2::ggplot() +  
+   geom_sf(data = brazil) +  
+   geom_sf(data = coord_geo %>%  
+     dplyr::filter(!is.na(capital)), color = 'red') +  
+   geom_sf_label(data = coord_geo %>%  
+     dplyr::filter(population_proper >= 150000),  
+     aes(label = city), size = 3, hjust = 0)
```

```
R> brplot
```

Learn More

Sources
