Reading Spatial Data in R

HES 505 Fall 2023: Session 4

Matt Williamson

Objectives

- 1. Revisit the components of spatial data
- 2. Describe some of the key considerations for thinking about spatial data
- 3. Introduce the two primary **R** packages for spatial workflows
- 4. Learn to read and explore spatial objects in **R**

Describing Absolute Locations

- **Coordinates:** 2 or more measurements that specify location relative to a *reference system*
- Cartesian coordinate system
- *origin* (*O*) = the point at which both measurement systems intersect
- Adaptable to multiple dimensions (e.g. *z* for altitude)



Cartesian Coordinate System

Locations on a Globe

• The earth is not flat...



Latitude and Longitude

Locations on a Globe

- The earth is not flat...
- Global Reference Systems (GRS)
- *Graticule*: the grid formed by the intersection of longitude and latitude
- The graticule is based on an ellipsoid model of earth's surface and contained in the *datum*

Global Reference Systems

The *datum* describes which ellipsoid to use and the precise relations between locations on earth's surface and Cartesian coordinates

- Geodetic datums (e.g., WGS84): distance from earth's center of gravity
- Local data (e.g., NAD83): better models for local variation in earth's surface

Describing location: extent

- How much of the world does the data cover?
- For rasters, these are the corners of the lattice
- For vectors, we call this the bounding box

Describing location: resolution

- Resolution: the accuracy that the location and shape of a map's features can be depicted
- Minimum
 Mapping Unit: The minimum size and dimensions that can
 be reliably
 represented at a
 given map scale.
- Map scale vs. scale of analysis



The earth is not flat...

Projections

- But maps, screens, and publications are...
- **Projections** describe *how* the data should be translated to a flat surface
- Rely on 'developable surfaces'
- Described by the Coordinate Reference System (CRS)



Developable Surfaces

Projection necessarily induces some form of distortion (tearing, compression, or shearing)

Coordinate Reference Systems

- Some projections minimize distortion of angle, area, or distance
- Others attempt to avoid extreme distortion of any kind
- Includes: Datum, ellipsoid, units, and other information (e.g., False Easting, Central Meridian) to further map the projection to the GCS
- Not all projections have / require all of the parameters



Choosing Projections



Nell Kaye/@neilrkaye

The Mercator Map Projection with the true size and shape of the country overlaid.

- Equal-area for thematic maps
- Conformal for presentations
- Mercator or equidistant for navigation and distance

Geometries, support, and spatial messiness

Geometries

- Vectors store

 aggregate the
 locations of a feature
 into a geometry
- Most vector

 operations require
 simple, valid
 geometries



Image Source: Colin Williams (NEON)

Valid Geometries

- A linestring is *simple* if it does not intersect
- Valid polygons
- Are closed (i.e., the last vertex equals the first)
- Have holes (inner rings) that inside the the exterior boundary
- Have holes that touch the exterior at no more than one vertex (they don't extend across a line) For multipolygons, adjacent polygons touch only at points
- Do not repeat their own path



Empty Geometries

- Empty geometries arise when an operation produces
 NULL outcomes (like looking for the intersection
 between two non-intersecting polygons)
- **sf** allows empty geometries to make sure that information about the data type is retained
- Similar to a data.frame with no rows or a list with NULL values
- Most vector operations require simple, valid geometries



• **Support** is the area to which an attribute applies.

Spatial Messiness

- Quantitative geography requires that our data are aligned
- Achieving alignment is part of reproducible workflows
- Making principled decisions about projections, resolution, extent, etc

Mapping Location in R

Data Types and R Packages

- Vector Data
 - Point features
 - Line features
 - Area features (polygons)
- Raster Data
 - Spatially continuous field
 - Based on pixels (not points)





Reading in Spatial Data: spreadsheets

- Most basic form of spatial data
- Need **x** (longitude) and **y** (latitude) as columns
- Need to know your CRS
- **read_***** necessary to bring in the data

```
library(tidyverse)
 1
   library(sf)
 2
3
   file.to.read <- read csv(file = "path/to/your/file",</pre>
 4
 5
                               col names = TRUE, col types = NULL,
                               na = na = c("", "NA"))
 6
 7
   file.as.sf <- st as sf(file.to.read,
 8
                            coords = c("longitude", "latitude"),
 9
                            crs=4326)
10
```

Reading in Spatial Data: shapefiles

Reading in Spatial Data: shapefiles

1 library(sf)

3

2 shapefile.inR <- read_sf(dsn = "path/to/file.shp",</pre>

layer=NULL, geometry_colu



Reading in Spatial Data: rasters

- rast will read rasters using the terra package
- Also used to create rasters from scratch
- Returns SpatRaster object

```
1 library(sf)
2 raster.inR <- rast(x = "path/to/file.shp",
3 lyrs=NULL)</pre>
```



Introducing the Data

- Good idea to get to know your data before manipulating it
- str, summary, nrow, ncol are good places to start
- st_crs (for sf class objects) and crs (for SpatRaster objects)
- We'll practice a few of these now...

Saving your data

 write_sf for sf objects; writeRaster for SpatRasters

```
1 library(sf)
```

```
2 library(terra)
```

3

```
4 write_sf(object = object.to.save, dsn = "path/to/save/object", append = FAL
```

```
5 writeRaster(x=object, filename = "path/to/save")
```