Vector Operations Part 1I

HES 505 Fall 2022: Session 9

Matt Williamson

Your final project

- At least 5 datasets total (1 tabular, 1 vector, 1 raster, and 2 of your choosing)
- Choose 1 statistical approach to address your research question
- Visualizations - minimum of 3. 1 location map; the others should help address your question
- Submission formats
- A note about your discussion

Today’s Plan

E 47th St
47th Street (Kenwood)
South Michigan Avenue
S Indiana Avenue
47th
51st
Garfield
E Hyde Park Blvd
51st / 53rd Street (Hyde Park)
E 53rd St
E 55th St
55th - 56th - 57th Street
E Garfield Blvd
Washington Park
Morgan Dr
E 57th St
59th Street, University of Chicago
Harvard Elementary School
Kenwood Academy
DuSable High School
Meadville Theological School
Chicago Theological Seminary

Objectives

By the end of today, you should be able to:

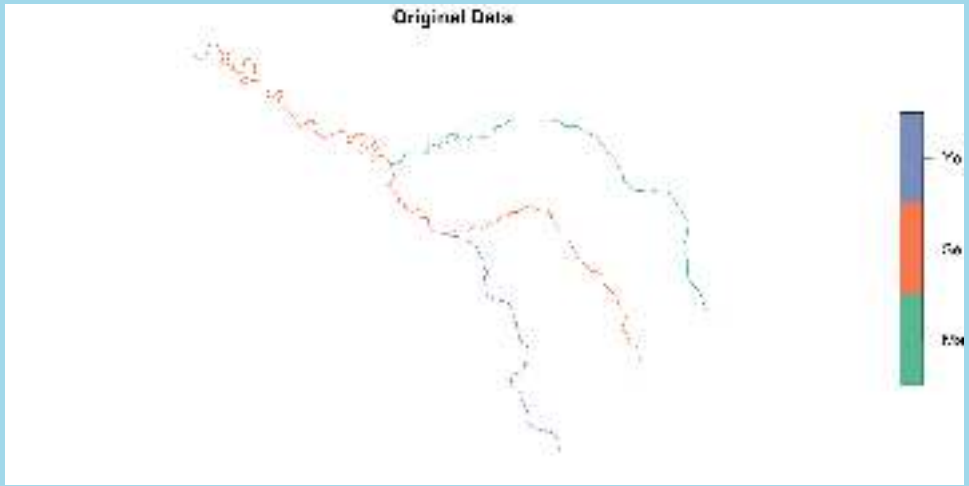
- Complete a workflow for identifying and remedying invalid geometries
- Describe the various unary, binary, and n-ary transformers
- Use predicates and `dplyr::filter` to subset spatial data

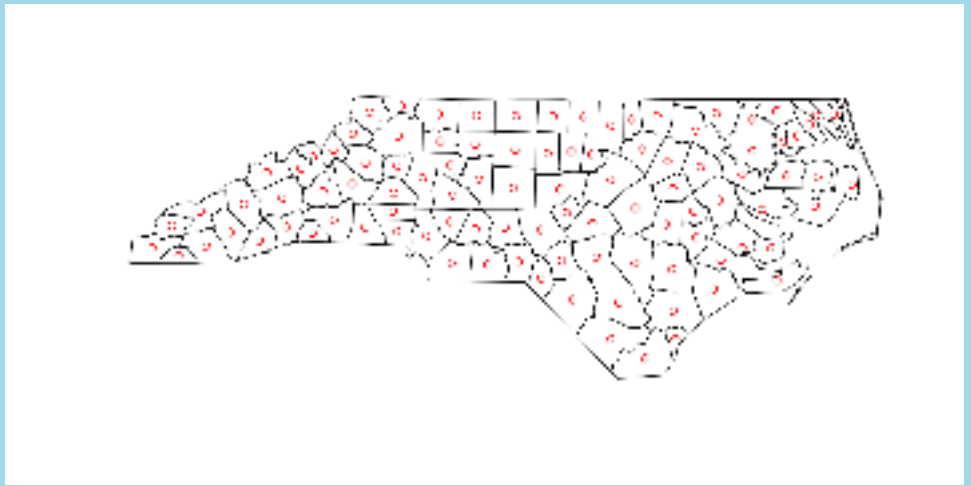
Revisiting `predicates` and `measures`

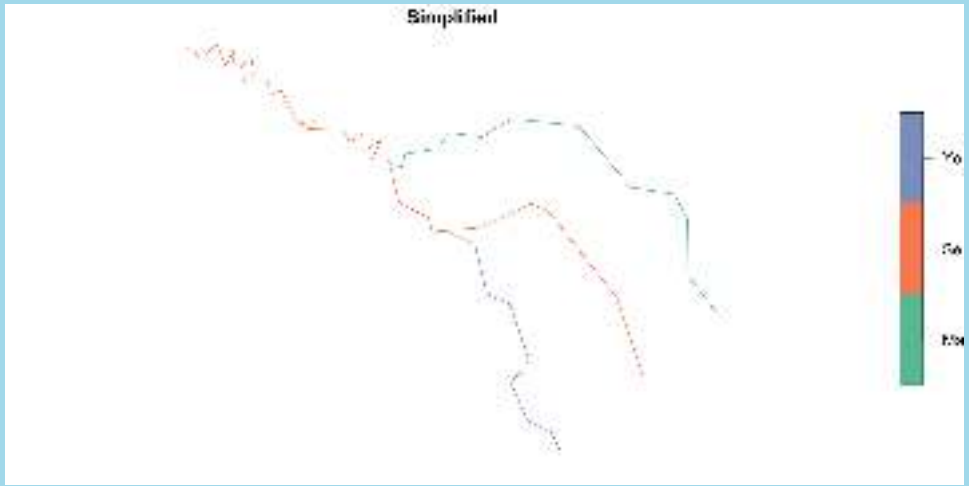
- **Predicates**: evaluate a logical statement asserting that a property is `TRUE`
- **Measures**: return a numeric value with units based on the units of the CRS
- Unary, binary, and n-ary distinguish how many geometries each function accepts and returns

Transformations

- **Transformations**: create new geometries based on input geometries


Original Data




Simplified

Unary Transformations

transformer	returns a geometry …
`centroid`	of type `POINT` with the geometry's centroid
`buffer`	that is this larger (or smaller) than the input geometry, depending on the buffer size
`jitter`	that was moved in space a certain amount, using a bivariate uniform distribution
`wrap_dateline`	cut into pieces that do no longer cover the dateline
`boundary`	with the boundary of the input geometry
`convex_hull`	that forms the convex hull of the input geometry
`line_merge`	after merging connecting `LINESTRING` elements of a `MULTILINESTRING` into longer `LINESTRING`s.
`make_valid`	that is valid
`node`	with added nodes to linear geometries at intersections without a node; only works on individual linear geometries
`point_on_surface`	with a (arbitrary) point on a surface
`polygonize`	of type polygon, created from lines that form a closed ring

Unary Transformations (cont'd)

transformer	returns a geometry …
`segmentize`	a (linear) geometry with nodes at a given density or minimal distance
`simplify`	simplified by removing vertices/nodes (lines or polygons)
`split`	that has been split with a splitting linestring
`transform`	transformed or convert to a new coordinate reference system (chapter @ref(cs))
`triangulate`	with Delauney triangulated polygon(s) (figure @ref(fig:vor))
`voronoi`	with the Voronoi tessellation of an input geometry (figure @ref(fig:vor))
`zm`	with removed or added `Z` and/or `M` coordinates
`collection_extract`	with subgeometries from a `GEOMETRYCOLLECTION` of a particular type
`cast`	that is converted to another type
`+`	that is shifted over a given vector
`*`	that is multiplied by a scalar or matrix

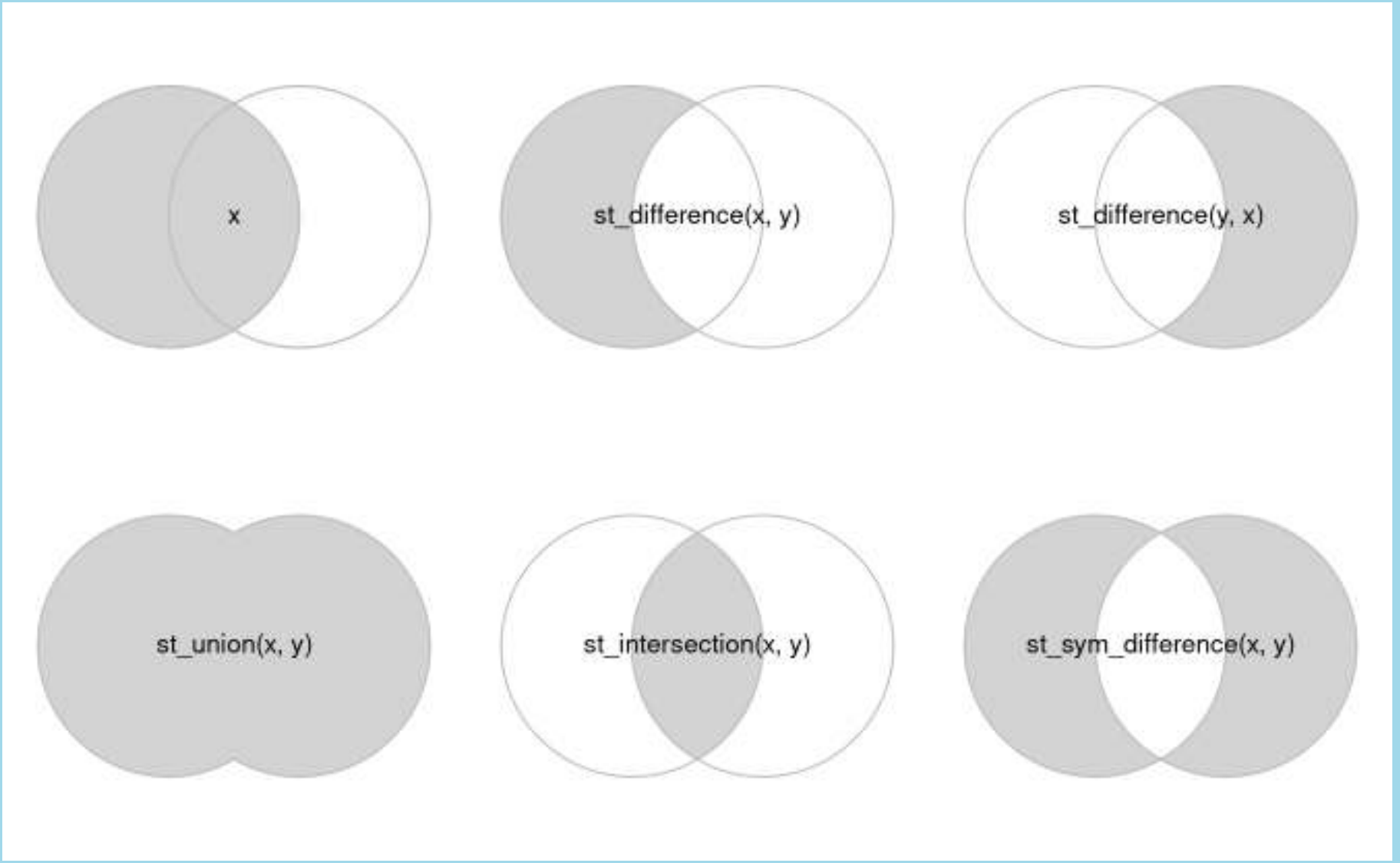
Common uses of Unary Transformers

- Creating valid geometries
- Reprojecting your data
- Combining or changing geometries

Binary Transformers

function	returns	infix operator
`intersection`	the overlapping geometries for pair of geometries	`&`
`union`	the combination of the geometries; removes internal boundaries and duplicate points, nodes or line pieces	`|`
`difference`	the geometries of the first after removing the overlap with the second geometry	`/`
`sym_difference`	the combinations of the geometries after removing where they intersect; the negation (opposite) of `intersection`	`%/%`
`crop`	crop an sf object to a specific rectangle	

Binary Transformers



Common Uses of Binary Transformers

- Relating partially overlapping datasets to each other
- Reducing the extent of vector objects

N-ary Transformers

- Similar to Binary (except `st_crop`)
- `union` can be applied to a set of geometries to return its geometrical union
- `intersection` and `difference` take a single argument, but operate (sequentially) on all pairs, triples, quadruples, etc.

Subsetting Data

Subsetting Data

- Often want to restrict analyses to particular locations
- Can combine `predicates` with `[]` to subset based on geography
- Can also use `dplyr::filter` and `dplyr::select` to subset using attributes

Using `predicates`

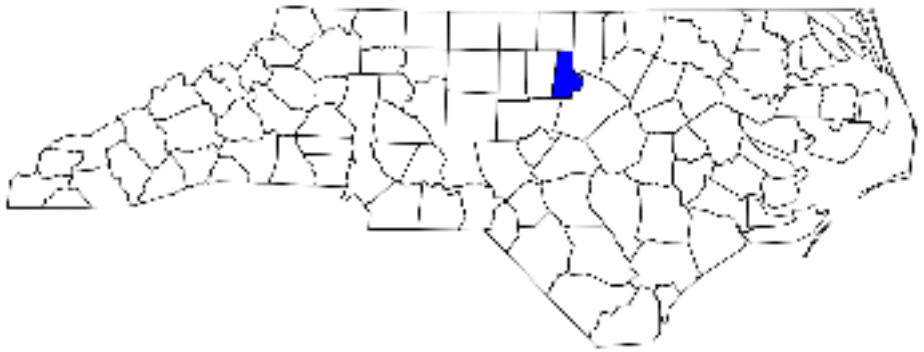
- Can combine `predicates` with `[]` to subset based on topological relations
- `x[y, , op = st_intersects]`
- `st_filter( x = x, y = y, .predicate = st_intersects)`

Using `dplyr`

- `filter` returns rows that match a criteria
- `select` returns columns

```r
library(tidyverse)
durham.cty <- nc %>%
  filter(., NAME == "Durham")
## We can also use the bracket approach
durham.cty2 <- nc[nc$NAME == "Durham",]

plot(st_geometry(nc))
plot(st_geometry(durham.cty), add=TRUE, cc
```



```r
nc.select <- nc %>%
  select(., c("BIR79", "SI
plot(nc.select)
```