Spatial data with sf

Programming for Statistical Science

Shawn Santo

Supplementary materials

Full video lecture available in Zoom Cloud Recordings

Additional resources

- Simple Features for R vignettes
- CRS in R by Melanie Frazier
- Leaflet for R

Introduction

Spatial data is different

Our **typical tidy data frame**:

```
#> # A tibble: 336,776 x 19
#>     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
#>    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
#>  1  2013     1     1      517            515         2      830            819
#>  2  2013     1     1      533            529         4      850            830
#>  3  2013     1     1      542            540         2      923            850
#>  4  2013     1     1      544            545        -1     1004           1022
#>  5  2013     1     1      554            600        -6      812            837
#>  6  2013     1     1      554            558        -4      740            728
#>  7  2013     1     1      555            600        -5      913            854
#>  8  2013     1     1      557            600        -3      709            723
#>  9  2013     1     1      557            600        -3      838            846
#> 10  2013     1     1      558            600        -2      753            745
#> # … with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
#> #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
#> #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

Spatial data is different

A **simple features object**:

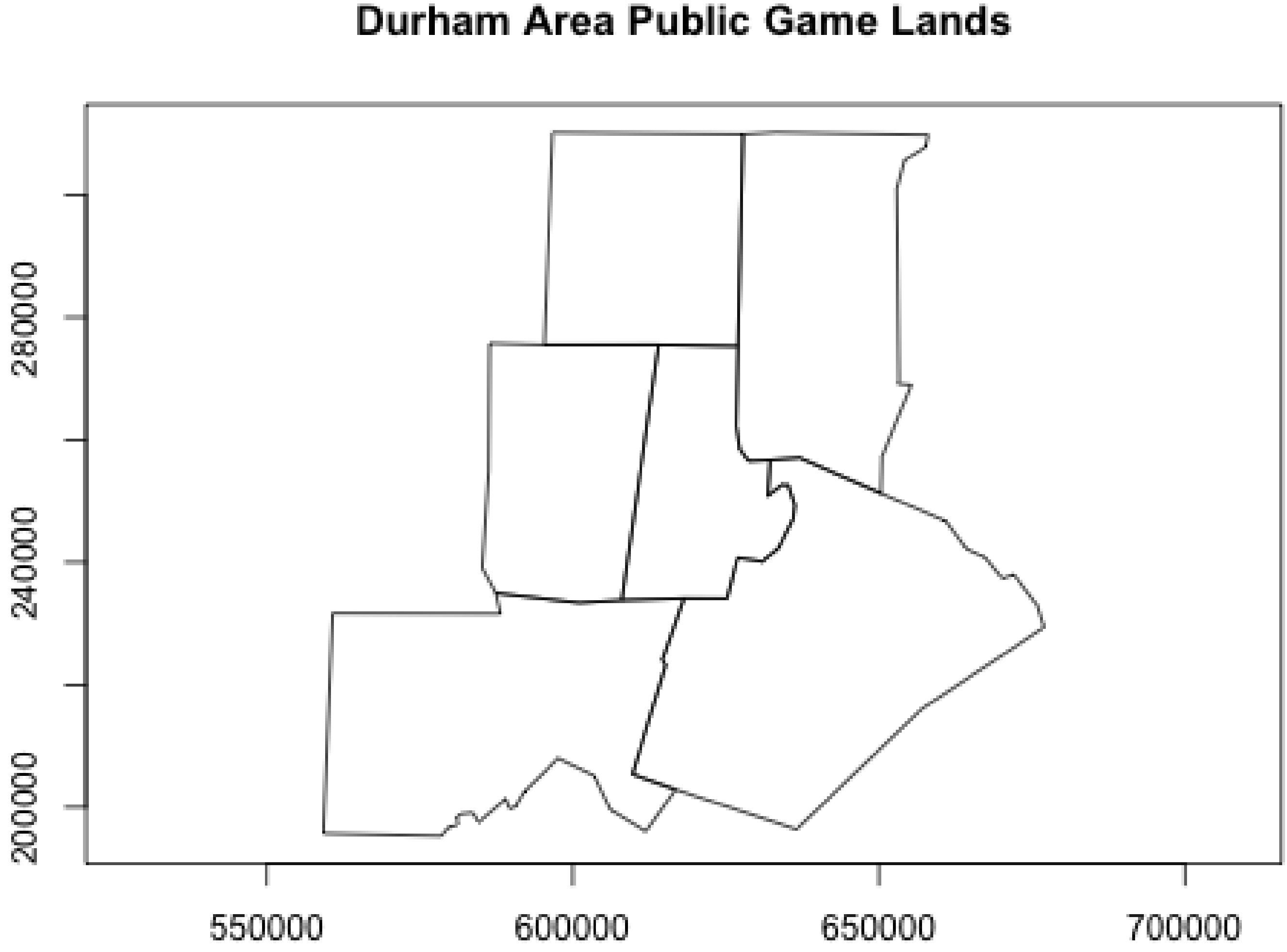
```
#> Simple feature collection with 100 features and 5 fields
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
#> geographic CRS: NAD27
#> First 10 features:
#>     AREA PERIMETER CNTY_ CNTY_ID        NAME                       geometry
#> 1  0.114     1.442  1825    1825        Ashe MULTIPOLYGON (((-81.47276 3...
#> 2  0.061     1.231  1827    1827   Alleghany MULTIPOLYGON (((-81.23989 3...
#> 3  0.143     1.630  1828    1828       Surry MULTIPOLYGON (((-80.45634 3...
#> 4  0.070     2.968  1831    1831   Currituck MULTIPOLYGON (((-76.00897 3...
#> 5  0.153     2.206  1832    1832 Northampton MULTIPOLYGON (((-77.21767 3...
#> 6  0.097     1.670  1833    1833    Hertford MULTIPOLYGON (((-76.74506 3...
#> 7  0.062     1.547  1834    1834      Camden MULTIPOLYGON (((-76.00897 3...
#> 8  0.091     1.284  1835    1835       Gates MULTIPOLYGON (((-76.56251 3...
#> 9  0.118     1.421  1836    1836      Warren MULTIPOLYGON (((-78.30876 3...
#> 10 0.124     1.428  1837    1837      Stokes MULTIPOLYGON (((-80.02567 3...
```

Another simple features object:

```
#> Simple feature collection with 94 features and 1 field
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: 127456.7 ymin: 26544.91 xmax: 923528.7 ymax: 318097.4
#> projected CRS:  NAD83 / North Carolina
#> # A tibble: 94 x 2
#>    GML_HAB                                                              geometry
#>    <chr>                                                      <MULTIPOLYGON [m]>
#>  1 Alcoa               (((512096.2 183241.7, 512185.7 183203.4, 512226 183186.2…
#>  2 Alligator River     (((869633.1 244541.9, 869739.4 243987.6, 869762.7 243999…
#>  3 Angola Bay          (((713079.4 113954.7, 713110.9 113878.7, 713133.1 113925…
#>  4 Bachelor Bay        (((813742.2 238618.7, 813730 238603.2, 813693.8 238525.7…
#>  5 Bertie County       (((797133.8 247034.5, 797119.5 247030, 797112.2 247027.7…
#>  6 Bladen Lakes State… (((658970.6 95406.32, 660025.1 94245.76, 659839.4 94144.…
#>  7 Brinkleyville       (((714741 276970.3, 714623.9 276970, 714622.1 277000, 71…
#>  8 Buckhorn            (((589723.7 253224.6, 589568.5 252937.2, 589689.8 252937…
#>  9 Buckridge           (((871137.4 219894.9, 871124.9 219827.8, 871124.2 219828…
#> 10 Buffalo Cove        (((381445.9 260375.4, 381574.9 259668.3, 381915 259796.3…
#> # … with 84 more rows
```

Spatial data plotting needs care

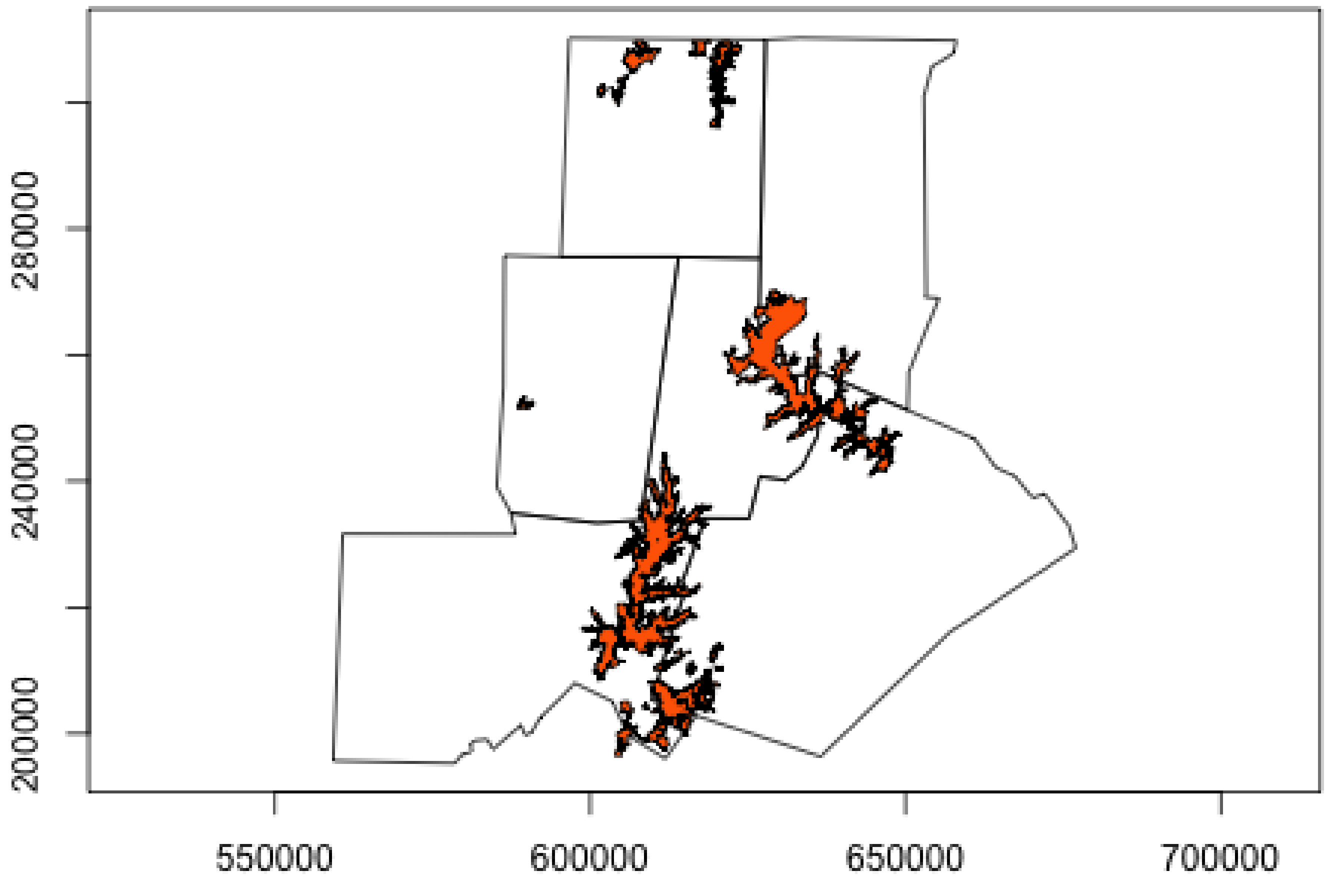
Can we combine the two plots?



Where are the game lands?

We can, but more care is needed.

Durham Area Public Game Lands



Spatial data challenges

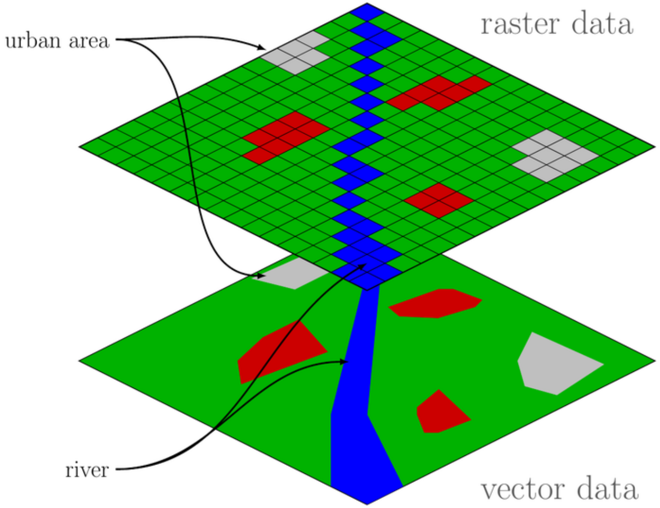
1. Different data types exist.
2. Special attention must be given to the coordinate reference system (CRS).
3. Manipulating spatial data objects is similar but not identical to manipulating data frame objects.

Spatial data and R

Analysis of spatial data in R

- Package raster contains classes and tools for handling spatial raster data.

- Package sf combines the functionality of sp, rgdal, and rgeos into a single package based on tidy simple features.



Whether or not you use vector or raster data depends on the type of problem and the data source. Our focus will be on vector data and package sf.

Source: https://commons.wikimedia.org/wiki/File:Raster_vector_tikz.png

Installing package `sf`

From https://r-spatial.github.io/sf/index.html

Windows

Installing `sf` from source works under windows when Rtools is installed. This downloads the system requirements from rwinlib.

MacOS

```
brew install pkg-config
brew install gdal
```

Once gdal is installed, you will be able to install sf package from source in R.

Linux

For Unix-alikes, GDAL (>= 2.0.1), GEOS (>= 3.4.0) and Proj.4 (>= 4.8.0) are required.

Features and simple features

- A **feature** is a thing or object in the real world: a house, a city, a park, a forest, etc.
- A **simple feature** as defined by OpenGIS Abstract specification is to have both spatial and non-spatial attributes. Spatial attributes are geometry valued, and simple features are based on 2D geometry with linear interpolation between vertices.

```
Simple feature collection with 100 features and 1 field
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 123829.8 ymin: 14740.06 xmax: 930518.6 ymax: 318255.5
projected CRS:  NAD83 / North Carolina
First 10 features:
          NAME                       geometry
1         Ashe MULTIPOLYGON (((387344.7 27...
2    Alleghany MULTIPOLYGON (((408601.4 29...
3        Surry MULTIPOLYGON (((478715.7 27...
4    Currituck MULTIPOLYGON (((878193.4 28...
5  Northampton MULTIPOLYGON (((769834.9 27...
6     Hertford MULTIPOLYGON (((812327.7 27...
7       Camden MULTIPOLYGON (((878193.4 28...
8        Gates MULTIPOLYGON (((828444.5 29...
9       Warren MULTIPOLYGON (((671746.3 27...
10      Stokes MULTIPOLYGON (((517435.1 27...
```

Simple features examples

sf objects

```
nc <- st_read(system.file("shape/nc.shp", package = "sf"), quiet = TRUE)
nc
```

```
#> Simple feature collection with 100 features and 14 fields
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
#> geographic CRS: NAD27
#> First 10 features:
#>     AREA PERIMETER CNTY_ CNTY_ID        NAME  FIPS FIPSNO CRESS_ID BIR74 SID74
#> 1  0.114     1.442  1825    1825        Ashe 37009  37009        5  1091     1
#> 2  0.061     1.231  1827    1827   Alleghany 37005  37005        3   487     0
#> 3  0.143     1.630  1828    1828       Surry 37171  37171       86  3188     5
#> 4  0.070     2.968  1831    1831   Currituck 37053  37053       27   508     1
#> 5  0.153     2.206  1832    1832 Northampton 37131  37131       66  1421     9
#> 6  0.097     1.670  1833    1833    Hertford 37091  37091       46  1452     7
#> 7  0.062     1.547  1834    1834      Camden 37029  37029       15   286     0
#> 8  0.091     1.284  1835    1835       Gates 37073  37073       37   420     0
#> 9  0.118     1.421  1836    1836      Warren 37185  37185       93   968     4
#> 10 0.124     1.428  1837    1837      Stokes 37169  37169       85  1612     1
#>    NWBIR74 BIR79 SID79 NWBIR79                       geometry
#> 1       10  1364     0      19 MULTIPOLYGON (((-81.47276 3...
#> 2       10   542     3      12 MULTIPOLYGON (((-81.23989 3...
#> 3      208  3616     6     260 MULTIPOLYGON (((-80.45634 3...
#> 4      123   830     2     145 MULTIPOLYGON (((-76.00897 3...
#> 5     1066  1606     3    1197 MULTIPOLYGON (((-77.21767 3...
#> 6      954  1838     5    1237 MULTIPOLYGON (((-76.74506 3...
#> 7      115   350     2     139 MULTIPOLYGON (((-76.00897 3...
#> 8      254   594     2     371 MULTIPOLYGON (((-76.56251 3...
#> 9      748  1190     2     844 MULTIPOLYGON (((-78.30876 3...
#> 10     160  2038     5     176 MULTIPOLYGON (((-80.02567 3...
```

Class and other attributes: `sf`

```
class(nc)
```

```
#> [1] "sf"         "data.frame"
```

```
names(attributes(nc))
```

```
#> [1] "names"     "row.names" "class"     "sf_column" "agr"
```

`sfc` objects

```
nc_polygons <- st_geometry(nc)
nc_polygons
```

```
#> Geometry set for 100 features
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
#> geographic CRS: NAD27
#> First 5 geometries:
```

Class and other attributes: `sfc`

```
class(nc_polygons)
```

```
#> [1] "sfc_MULTIPOLYGON" "sfc"
```

```
names(attributes(nc_polygons))
```

```
#> [1] "n_empty"   "crs"       "class"     "precision" "bbox"
```

We see that `nc` has a class attribute `sf`, and object `nc_polygons` has a class attribute `sfc`. What methods are available?

```r
methods(class = "sf")
```

```
#>  [1] [                     [[<-                  $<-                  
#>  [4] aggregate             anti_join             arrange              
#>  [7] as.data.frame         cbind                 coerce               
#> [10] dbDataType            dbWriteTable          distinct             
#> [13] dplyr_reconstruct     filter                full_join            
#> [16] gather                group_by              group_split          
#> [19] identify              initialize            inner_join           
#> [22] left_join             mapView               merge                
#> [25] mutate                nest                  plot                 
#> [28] print                 rbind                 rename               
#> [31] right_join            sample_frac           sample_n             
#> [34] select                semi_join             separate_rows        
#> [37] separate              show                  slice                
#> [40] slotsFromS3           spread                st_agr               
#> [43] st_agr<-              st_area               st_as_s2             
#> [46] st_as_sf              st_bbox               st_boundary          
#> [49] st_buffer             st_cast               st_centroid          
#> [52] st_collection_extract st_convex_hull        st_coordinates       
#> [55] st_crop               st_crs                st_crs<-             
#> [58] st_difference         st_filter             st_geometry          
#> [61] st_geometry<-         st_interpolate_aw     st_intersection      
#> [64] st_intersects         st_is_valid           st_is                
#> [67] st_join               st_line_merge         st_m_range           
#> [70] st_make_valid         st_nearest_points     st_node              
#> [73] st_normalize          st_point_on_surface   st_polygonize        
#> [76] st_precision          st_reverse            st_sample            
#> [79] st_segmentize         st_set_precision      st_shift_longitude   
#> [82] st_simplify           st_snap               st_sym_difference    
#> [85] st_transform          st_triangulate        st_union             
#> [88] st_voronoi            st_wrap_dateline      st_write             
```

```r
methods(class = "sfc")
```

```
#>  [1] [                     [<-                   as.data.frame
#>  [4] c                     coerce                format
#>  [7] fortify               identify              initialize
#> [10] mapView               obj_sum               Ops
#> [13] print                 rep                   scale_type
#> [16] show                  slotsFromS3           st_area
#> [19] st_as_binary          st_as_grob            st_as_s2
#> [22] st_as_sf              st_as_text            st_bbox
#> [25] st_boundary           st_buffer             st_cast
#> [28] st_centroid           st_collection_extract st_convex_hull
#> [31] st_coordinates        st_crop               st_crs
#> [34] st_crs<-              st_difference         st_geometry
#> [37] st_intersection       st_intersects         st_is_valid
#> [40] st_is                 st_line_merge         st_m_range
#> [43] st_make_valid         st_nearest_points     st_node
#> [46] st_normalize          st_point_on_surface   st_polygonize
#> [49] st_precision          st_reverse            st_sample
#> [52] st_segmentize         st_set_precision      st_shift_longitude
#> [55] st_simplify           st_snap               st_sym_difference
#> [58] st_transform          st_triangulate        st_union
#> [61] st_voronoi            st_wrap_dateline      st_write
#> [64] st_z_range            st_zm                 str
#> [67] summary               type_sum              vec_cast.sfc
#> [70] vec_ptype2.sfc
#> see '?methods' for accessing help and source code
```

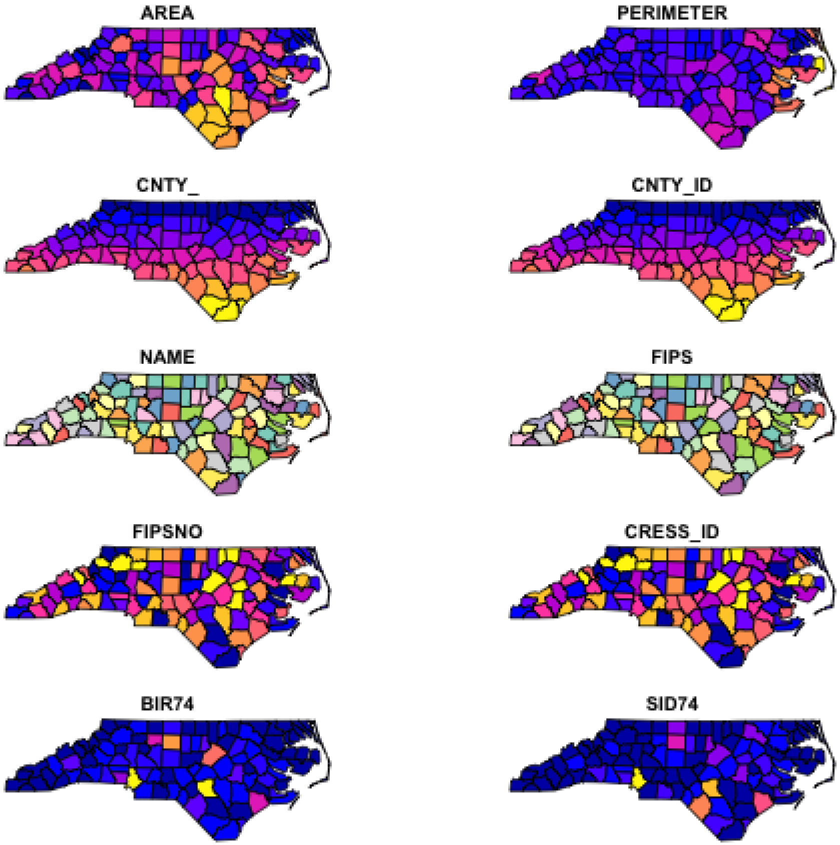
Reading and writing spatial data

- `st_read()` / `st_write()`, Shapefile, GeoJSON, KML, ...
- `st_as_sfc()`
- `st_as_text()`, well-known text format
- `st_as_binary()`, well-known binary format

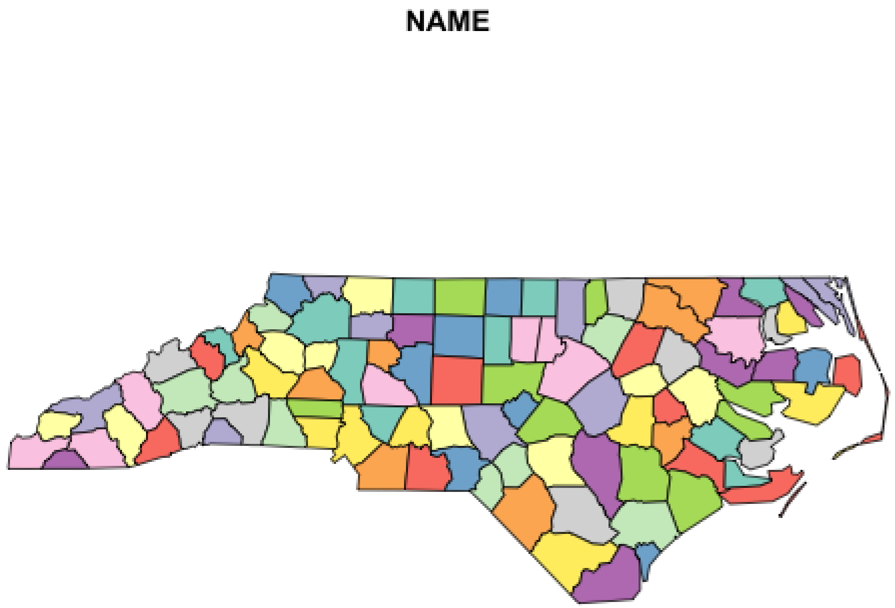
See https://r-spatial.github.io/sf/articles/sf2.html for the full set of driver availability.

Plotting with `plot()`

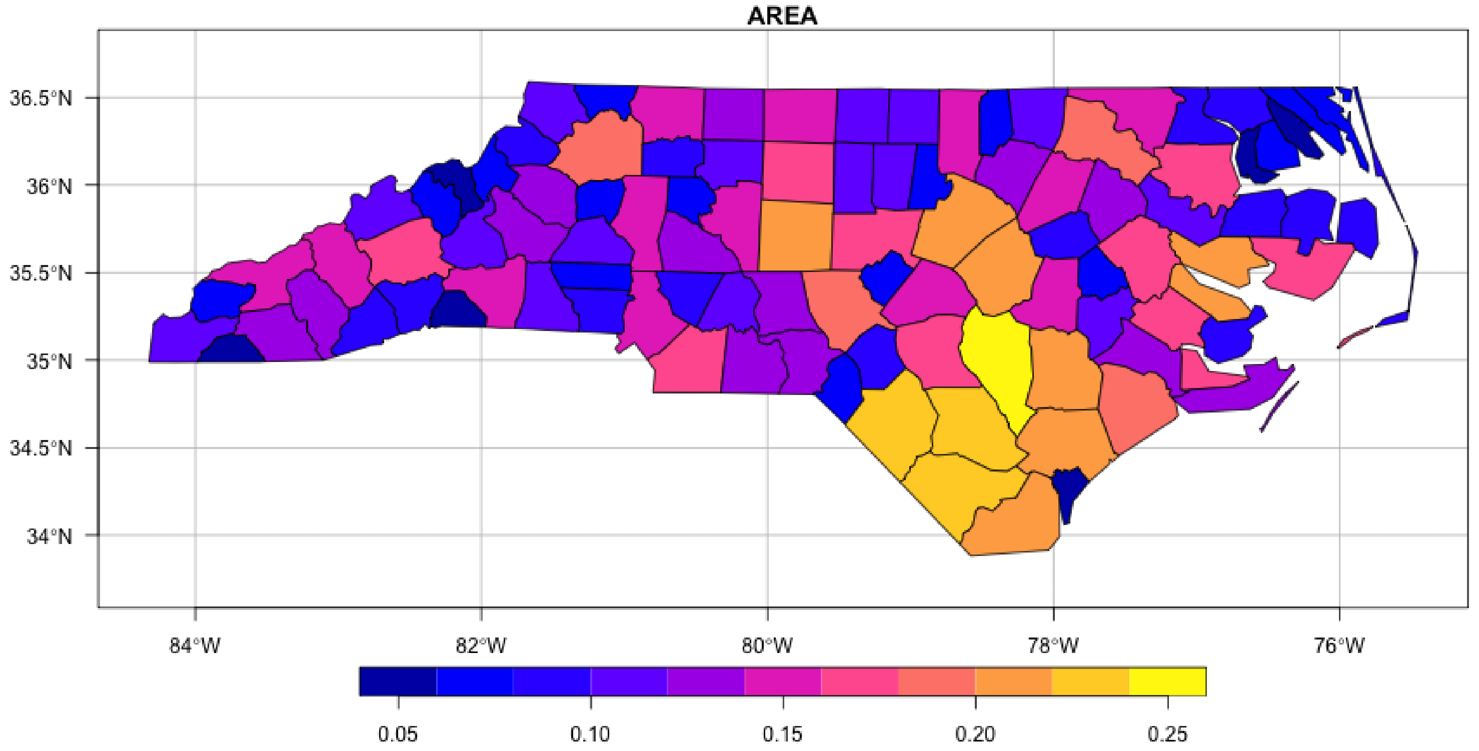
```
plot(nc)
```



```
plot(nc["NAME"])
```



```
par(oma=c(0,2,0,0))
plot(nc["AREA"], graticule = TRUE, axes = TRUE, las = 1)
```



What is happening with `[` and the `sf` object?

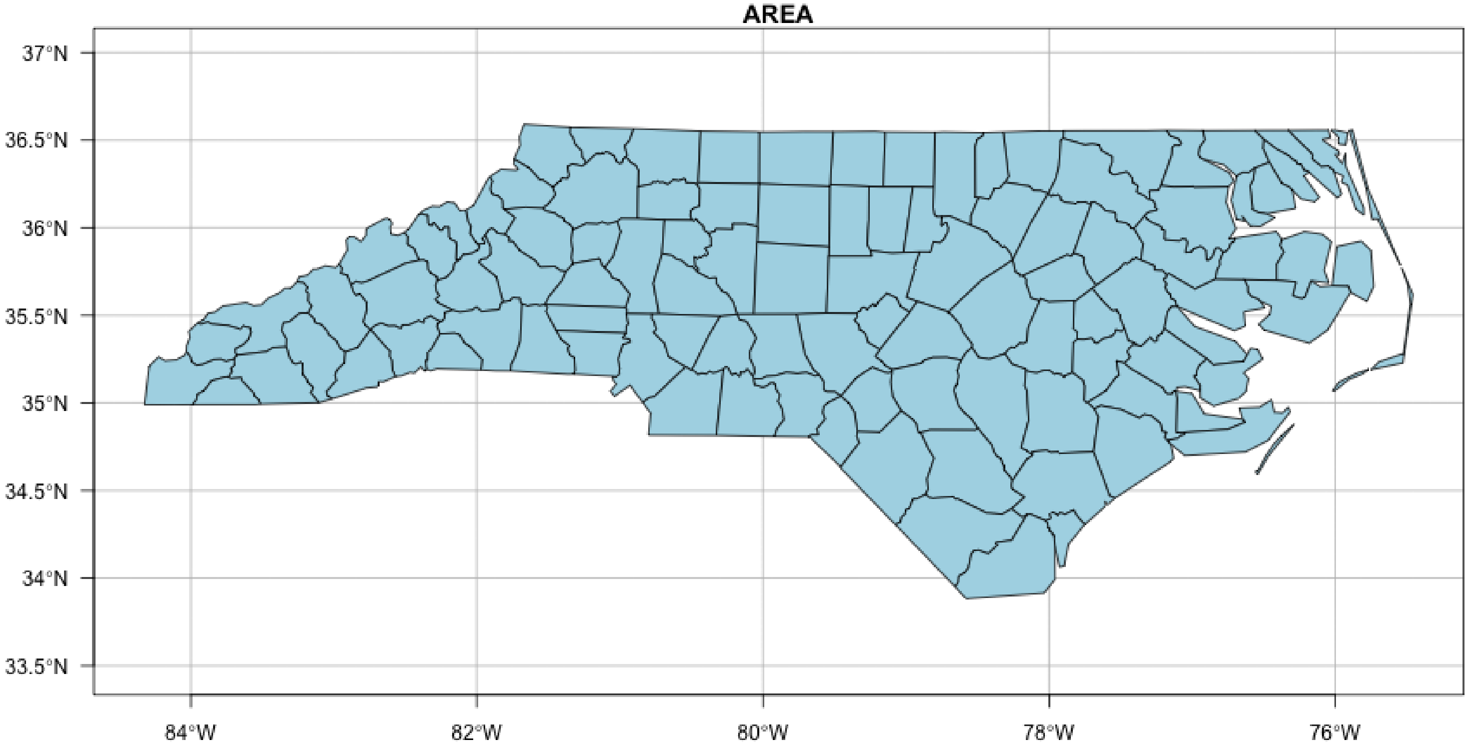
```
nc["AREA"]
```

```
#> Simple feature collection with 100 features and 1 field
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
#> geographic CRS: NAD27
#> First 10 features:
#>     AREA                       geometry
#> 1  0.114 MULTIPOLYGON (((-81.47276 3...
#> 2  0.061 MULTIPOLYGON (((-81.23989 3...
#> 3  0.143 MULTIPOLYGON (((-80.45634 3...
#> 4  0.070 MULTIPOLYGON (((-76.00897 3...
#> 5  0.153 MULTIPOLYGON (((-77.21767 3...
#> 6  0.097 MULTIPOLYGON (((-76.74506 3...
#> 7  0.062 MULTIPOLYGON (((-76.00897 3...
#> 8  0.091 MULTIPOLYGON (((-76.56251 3...
#> 9  0.118 MULTIPOLYGON (((-78.30876 3...
#> 10 0.124 MULTIPOLYGON (((-80.02567 3...
```

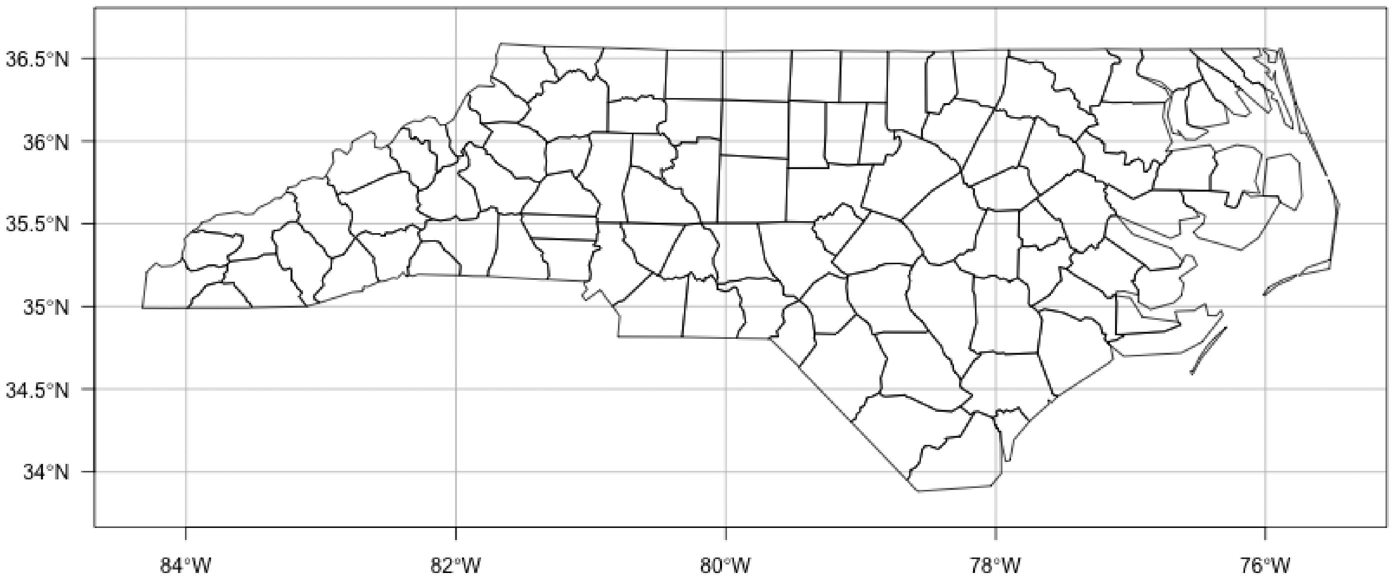
```r
nc$AREA
```

```
#>   [1] 0.114 0.061 0.143 0.070 0.153 0.097 0.062 0.091 0.118 0.124 0.114 0.153
#>  [13] 0.143 0.109 0.072 0.190 0.053 0.199 0.081 0.063 0.044 0.064 0.086 0.128
#>  [25] 0.108 0.170 0.111 0.180 0.104 0.077 0.142 0.059 0.131 0.122 0.080 0.118
#>  [37] 0.219 0.118 0.155 0.069 0.066 0.145 0.134 0.100 0.099 0.116 0.201 0.180
#>  [49] 0.094 0.134 0.168 0.106 0.168 0.207 0.144 0.094 0.203 0.141 0.070 0.065
#>  [61] 0.146 0.142 0.154 0.118 0.078 0.125 0.181 0.143 0.091 0.130 0.103 0.095
#>  [73] 0.078 0.104 0.098 0.091 0.060 0.131 0.241 0.082 0.120 0.172 0.121 0.163
#>  [85] 0.138 0.098 0.167 0.204 0.121 0.051 0.177 0.080 0.195 0.240 0.125 0.225
#>  [97] 0.214 0.240 0.042 0.212
```

```
par(oma=c(0,2,0,0))
plot(nc["AREA"], col = "lightblue", graticule = TRUE,
     axes = TRUE, las = 1)
```

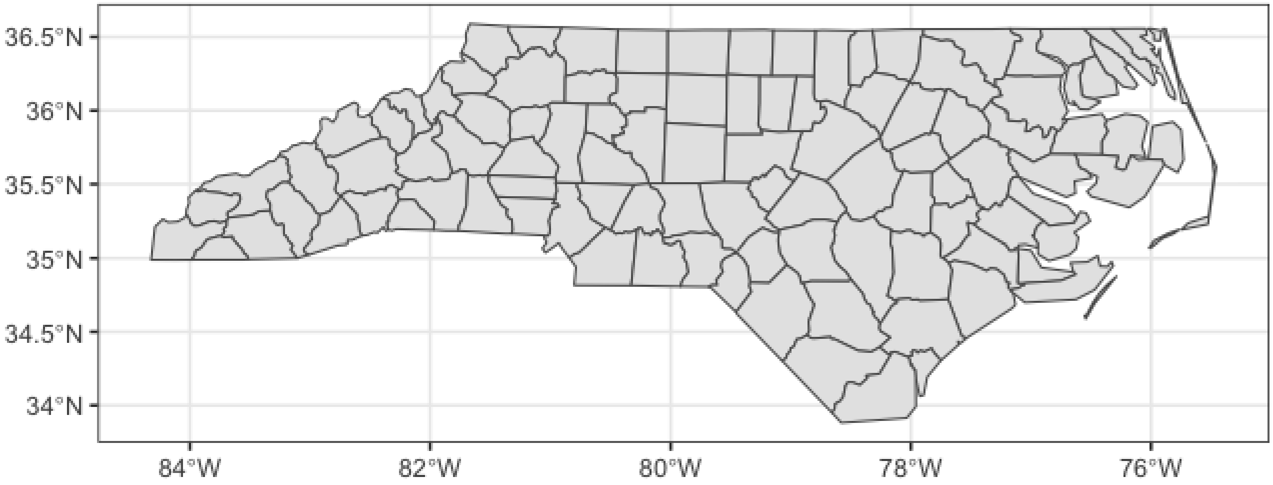


```
par(oma=c(0,2,0,0))
plot(st_geometry(nc), graticule = TRUE, axes = TRUE, las = 1)
```

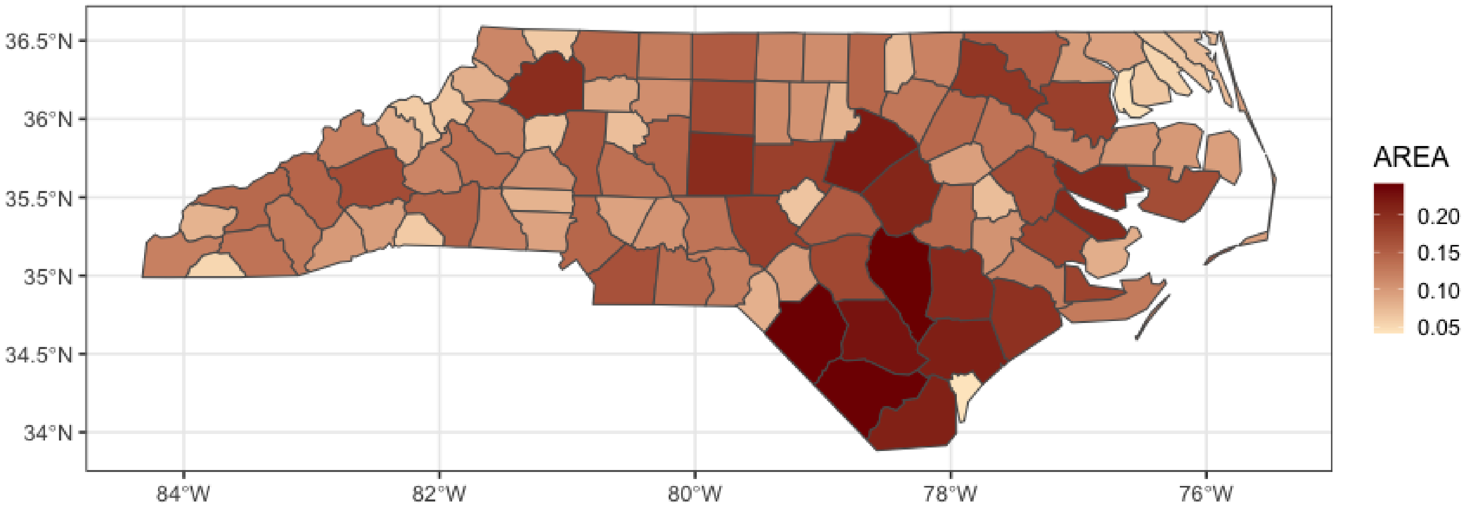


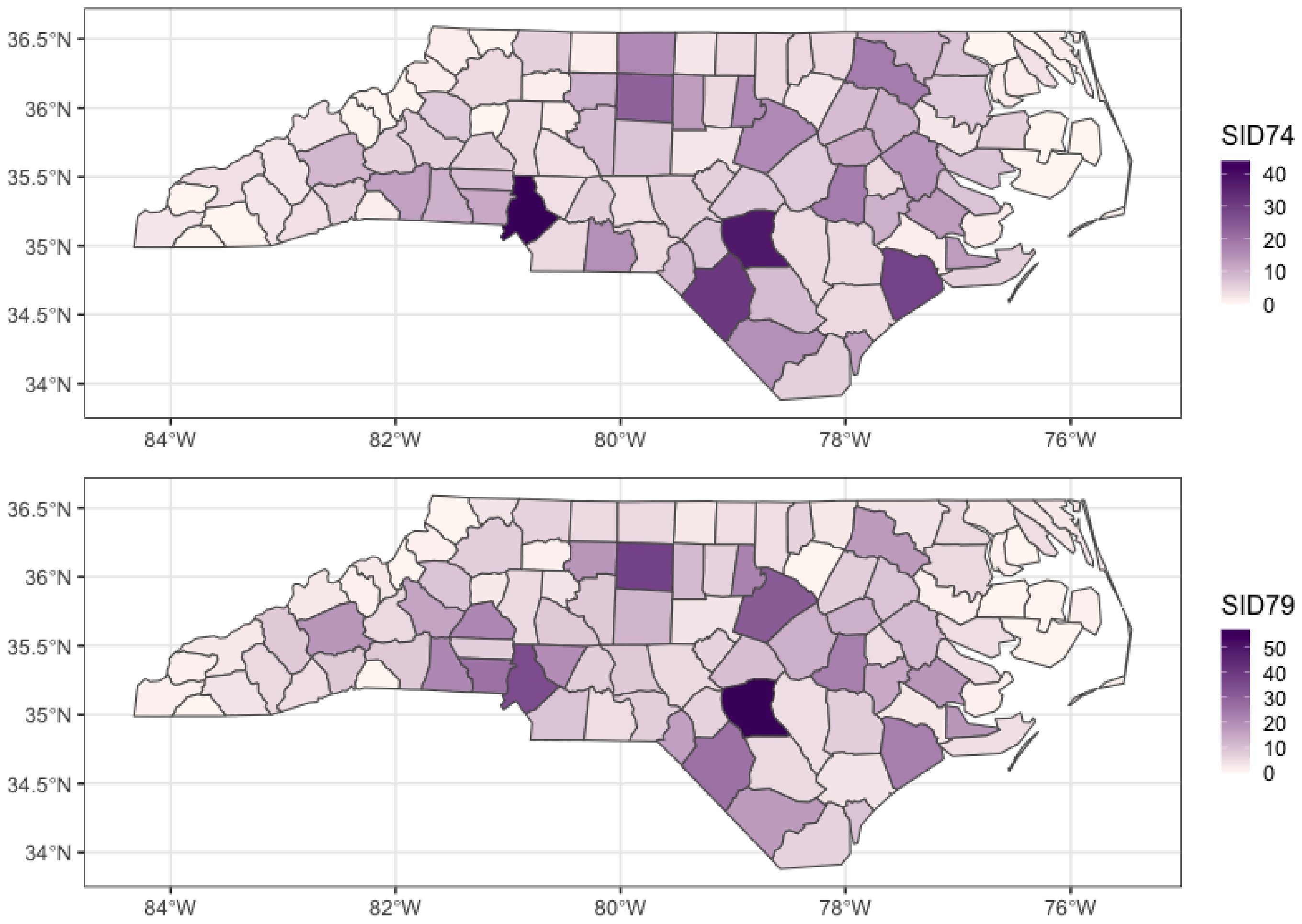
Plotting with `ggplot()`

```
ggplot(nc) +
  geom_sf() +
  theme_bw(base_size = 16)
```



```r
ggplot(nc) +
  geom_sf(aes(fill = AREA)) +
  scale_fill_gradient(low = "#fee8c8", high = "#7f0000") +
  theme_bw(base_size = 16)
```




36.5°N
36°N
35.5°N
35°N
34.5°N
34°N
84°W
82°W
80°W
78°W
76°W
SID74
40
30
20
10
0
SID79
50
40
30
20
10
0

```
p1 <- ggplot(nc) +
  geom_sf(aes(fill = SID74)) +
  scale_fill_gradient(low = "#fff7f3", high = "#49006a") +
  theme_bw(base_size = 16)

p2 <- ggplot(nc) +
  geom_sf(aes(fill = SID79)) +
  scale_fill_gradient(low = "#fff7f3", high = "#49006a") +
  theme_bw(base_size = 16)

p1 / p2
```

Visually, what is wrong with the last plot?

Plotting with `mapview()`

```
mapview(nc)
```

```
mapviewOptions(legend.pos = "bottomright")
mapview(nc["SID74"], col.regions = sf.colors(10),
        layer.name = "SID 1974")
```

These should run in RStudio. There is an issue embedding this overlay in the slides.

Exercise

Use `ggplot` to create a choropleth map for the proportion of sudden infant deaths, for the period of July 1, 1974 to June 30, 1979.

Map layers

Game Lands data

The North Carolina Department of Environment and Natural Resources, Wildlife Resources Commission and the NC Center for Geographic Information and Analysis has a shapefile data set available on all public Game Lands in NC.

https://www.nconemap.gov/datasets/e5ddff9b96204c6181be7c022e61d946_0

We can directly download and unzip the shapefile via

```
download.file("https://opendata.arcgis.com/datasets/e5ddff9b96204c6181be7
                destfile = "data/Gamelands.zip")
unzip("data/Gamelands.zip", exdir = "data/")
```

To see the available files

```
list.files(path = "data/", pattern = "Game_Lands*")
```

```
#> [1] "Game_Lands_-_general.cpg" "Game_Lands_-_general.dbf"
#> [3] "Game_Lands_-_general.prj" "Game_Lands_-_general.shp"
#> [5] "Game_Lands_-_general.shx" "Game_Lands_-_general.xml"
```

Read in the shapefile

```
nc_gamelands <- st_read("data/Game_Lands_-_general.shp", quiet = TRUE)
```

```
print(nc_gamelands, n = 5)
```

```
#> Simple feature collection with 94 features and 6 fields
#> geometry type:  MULTIPOLYGON
#> dimension:      XY
#> bbox:           xmin: 127456.7 ymin: 26544.91 xmax: 923528.7 ymax: 318097.4
#> projected CRS:  NAD83 / North Carolina
#> First 5 features:
#>   OBJECTID         GML_HAB SUM_ACRES GameLandID Shape__Are Shape__Len
#> 1        1           Alcoa 11109.559          1   44958790  438301.56
#> 2        2 Alligator River 24439.089          2   98901485  151120.16
#> 3        3      Angola Bay 34067.382          3  137865804   87094.49
#> 4        4    Bachelor Bay  2786.258          4   11275585   26613.27
#> 5        5   Bertie County  3881.466          5   15707735   67343.97
#>                          geometry
#> 1 MULTIPOLYGON (((512096.2 18...
#> 2 MULTIPOLYGON (((869633.1 24...
#> 3 MULTIPOLYGON (((713079.4 11...
#> 4 MULTIPOLYGON (((813742.2 23...
#> 5 MULTIPOLYGON (((797133.8 24...
```

Metadata for each `sf` object

nc:

```
Simple feature collection with 100 features and 14 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.5
geographic CRS: NAD27
```

nc_gamelands:

```
Simple feature collection with 94 features and 6 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 127456.7 ymin: 26544.91 xmax: 923528.7 ymax: 318097
projected CRS:  NAD83 / North Carolina
```

Check the CRS

```
st_crs(nc)
```

```
Coordinate Reference System:
  User input: NAD27
  wkt:
GEOGCRS["NAD27",
    DATUM["North American Datum 1927",
        ELLIPSOID["Clarke 1866",6378206.4,294.978698213898,
            LENGTHUNIT["metre",1]]],
    PRIMEM["Greenwich",0,
        ANGLEUNIT["degree",0.0174532925199433]],
    CS[ellipsoidal,2],
        AXIS["latitude",north,
            ORDER[1],
            ANGLEUNIT["degree",0.0174532925199433]],
        AXIS["longitude",east,
            ORDER[2],
            ANGLEUNIT["degree",0.0174532925199433]],
    ID["EPSG",4267]]
```

```
st_crs(nc_gamelands)
```

```
Coordinate Reference System:
  User input: NAD83 / North Carolina
  wkt:
PROJCRS["NAD83 / North Carolina",
    BASEGEOGCRS["NAD83",
        DATUM["North American Datum 1983",
            ELLIPSOID["GRS 1980",6378137,298.257222101,
                LENGTHUNIT["metre",1]]],
        PRIMEM["Greenwich",0,
            ANGLEUNIT["degree",0.0174532925199433]],
        ID["EPSG",4269]],
    CONVERSION["SPCS83 North Carolina zone (meters)",
        METHOD["Lambert Conic Conformal (2SP)",
            ID["EPSG",9802]],
        PARAMETER["Latitude of false origin",33.75,
            ANGLEUNIT["degree",0.0174532925199433],
            ID["EPSG",8821]],

                  ⋮

        PARAMETER["Northing at false origin",0,
            LENGTHUNIT["metre",1],
            ID["EPSG",8827]]],
    CS[Cartesian,2],
        AXIS["easting (X)",east,
            ORDER[1],
            LENGTHUNIT["metre",1]],
        AXIS["northing (Y)",north,
            ORDER[2],
            LENGTHUNIT["metre",1]],
    USAGE[
        SCOPE["unknown"],
        AREA["USA - North Carolina"],
        BBOX[33.83,-84.33,36.59,-75.38]],
    ID["EPSG",32119]]
```

Coordinate reference systems (CRS)

- CRS provide a standardized way of describing locations.
- Different CRS arise from various ways data were gathered, the locations, and purposes of the data.
- A CRS is comprised of
  - an ellipsoid, to define the earth's shape;
  - a datum, to define the origin and orientation of coordinate axes;
  - a projection, to go from 3D to 2D.
- It is important that you transform your spatial data to a common CRS before plotting.

Transform CRS

```
nc_gamelands <- st_transform(nc_gamelands, crs = st_crs(nc))
```

Check they are equal:

```
st_crs(nc) == st_crs(nc_gamelands)
```

```
#> [1] TRUE
```

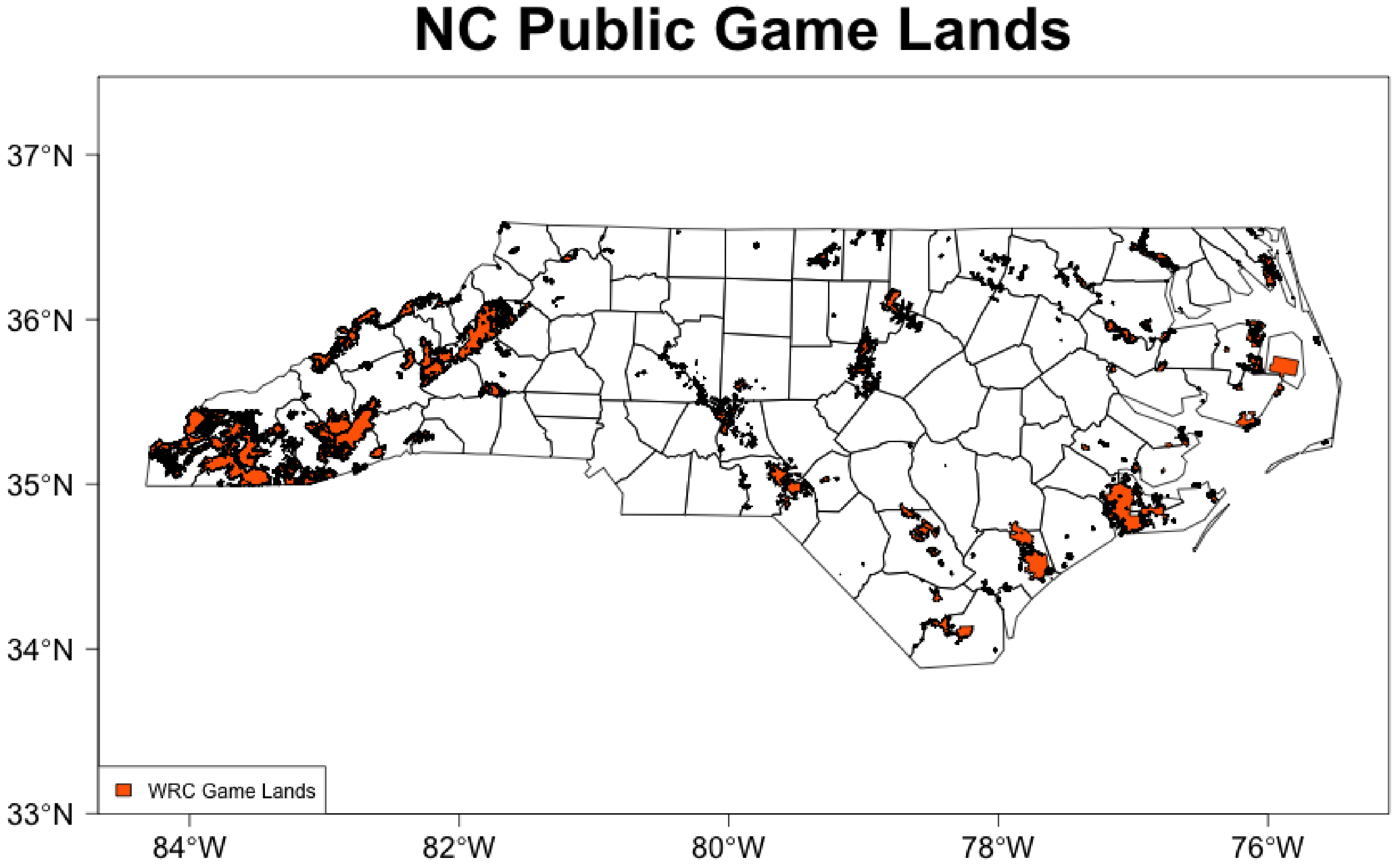
Map overlay with `plot()`

```r
plot(st_geometry(nc), axes = T, las = 1, main = "NC Public Game Lands",
     cex.main = 3, cex.lab = 2, cex.axis = 1.5)

plot(st_geometry(nc_gamelands),
     add = T,
     col = "#ff6700")

legend("bottomleft", legend = "WRC Game Lands", fill = "#ff6700")
```

Map overlay with `plot()`



Map overlay with `mapview()`

```r
nc_mapview <- mapview(nc, alpha.regions = .2, alpha = .9,
                      label = nc[, "NAME", drop = T],
                      layer.name = "NC Counties")
```

```r
nc_gamelands_mapview <- mapview(nc_gamelands, col.regions = "#ff6700",
         label = round(nc_gamelands[, "SUM_ACRES", drop = T], 2),
         layer.name = "NC Gamelands")
```

```r
nc_mapview + nc_gamelands_mapview
```

These should run in RStudio. There is an issue embedding this overlay in the slides.

Exercise

Create a map that includes NC county boundaries, Game Lands, and hazardous waste sites. Data for the hazardous waste sites is available at https://www.nconemap.gov/datasets/hazardous-waste-sites

This data set represents the location of sites within North Carolina that are regulated by the hazardous waste portions of the Resource Conservation and Recovery Act (RCRA).

Next time: Manipulating `sf` type objects