Welcome!

- Download the slide for today’s lab

- Open the following link in an internet browser:
  http://rpubs.com/bonwoodesign/CP6025_sf_2
  But don’t look at it for now

- Open R-Studio on a classroom computer or your own laptop

- When you are done with today’s lab activity, save your script and keep it for future references

Geospatial data in R - 2

12.2.2019

```
install.packages(‘sf’, dependencies = T)
install.packages(‘tmap’, dependencies = T)
```

If you are asked to type in y/n,
type in y

Studying for the exam

Because (I think) the final exam will be in a similar format as the assignments, the best ways to study include:

1. Getting the basic concepts right by going through the slides (and textbook if necessary).
2. Reading your assignment submissions and my comments (or answer_by_bonwoo.doc).
3. Preparing & familiarizing with R code and GeoDa for each task (e.g., OLS, logistic, etc.) so that you feel comfortable coding in the exam (you can bring your code to the exam).

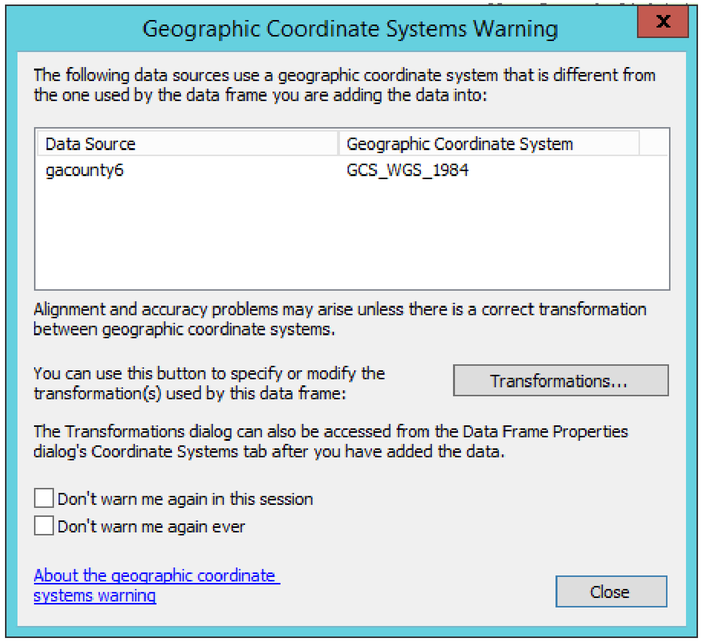
Coordinate Reference System

“.. map projections tries to transform the earth from its spherical shape (3D) to a planar shape (2D) … e.g., maps …

A **coordinate reference system** (CRS) then defines how the 2D, projected map in your GIS is related to real places on the earth” (QGIS Documentation, link in the note).

In order for select by location, spatial join, intersection, union, etc. to work between two (or more) sf objects, **you need to have them in the same coordinate reference system (crs).**

It is not just R; ArcMap does this too



Geographic Coordinate Systems Warning

The following data sources use a geographic coordinate system that is different from the one used by the data frame you are adding the data into:

Data Source	Geographic Coordinate System
gacounty6	GCS_WGS_1984

Alignment and accuracy problems may arise unless there is a correct transformation between geographic coordinate systems.

You can use this button to specify or modify the transformation(s) used by this data frame:

The Transformations dialog can also be accessed from the Data Frame Properties dialog's Coordinate Systems tab after you have added the data.

□ Don't warn me again in this session

□ Don't warn me again ever

About the geographic coordinate systems warning

Having the same CRS is like speaking the same language.

Coordinate Reference System

Changing CRS of a sf object:

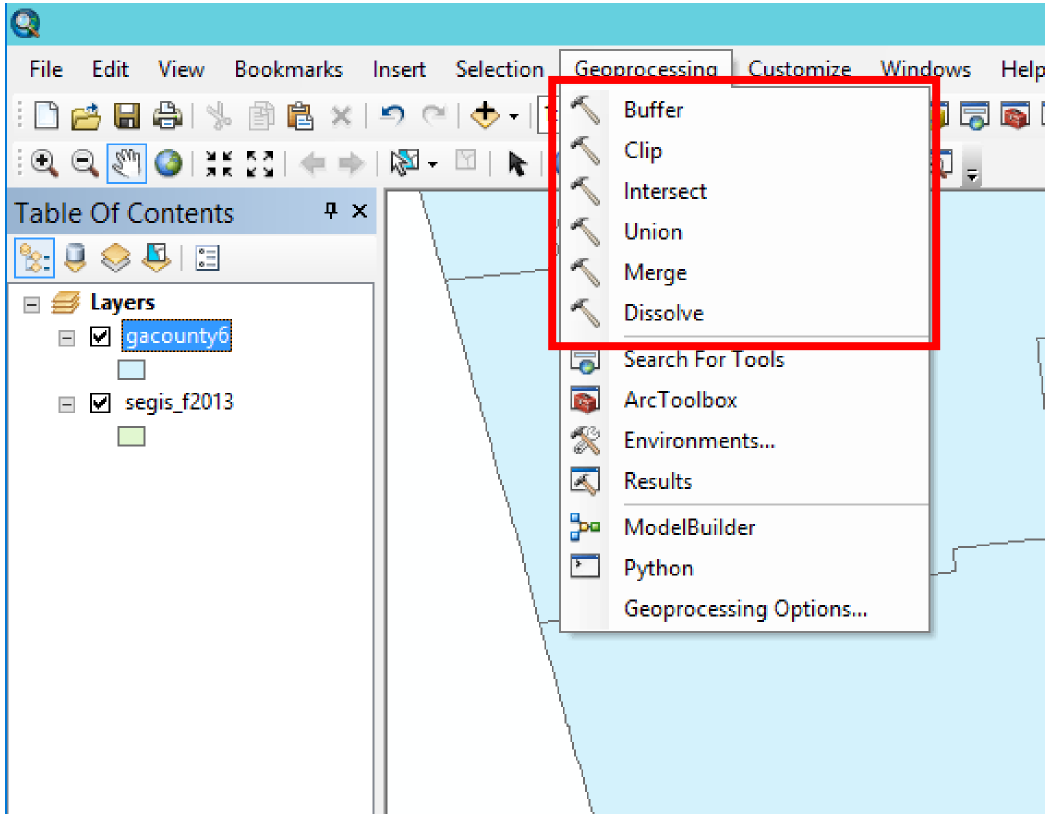
```
sf object  <- st_transform(sf object, crs = #####)
```

Commonly used CRS:

4326: WGS84 (geographic; unit in degrees; Google API uses it, but I may be wrong)

26967: NAD83 / Georgia West (projected; unit in meters)

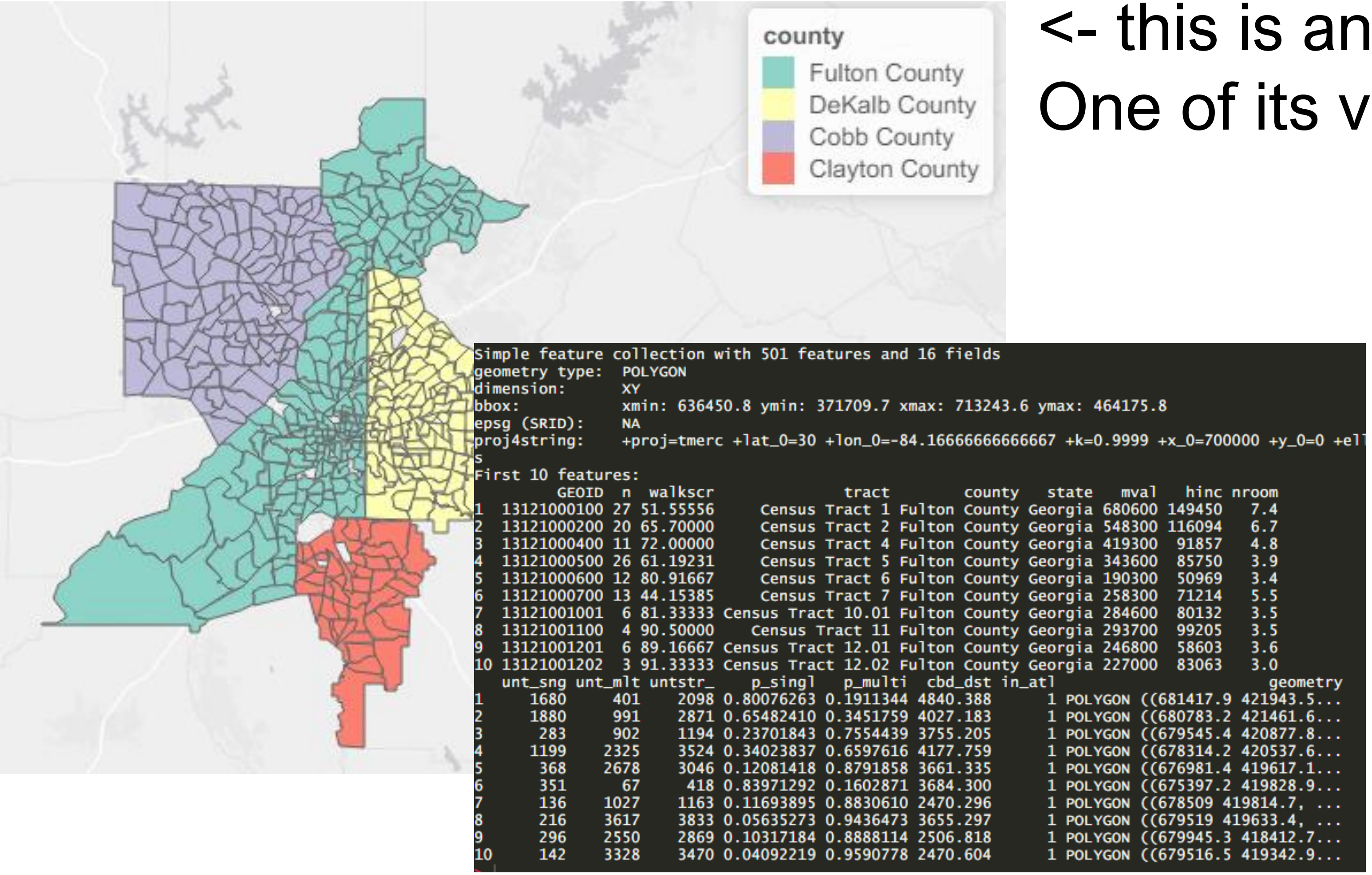
Geoprocessing



Buffer:	st_buffer
Clip:	st_intersection
Intersect:	st_intersection
Union:	st_union
Merge:	st_combine
Dissolve:	group_by + summarise

Make sure you unify the CRS of all the sf objects you want to operate on!

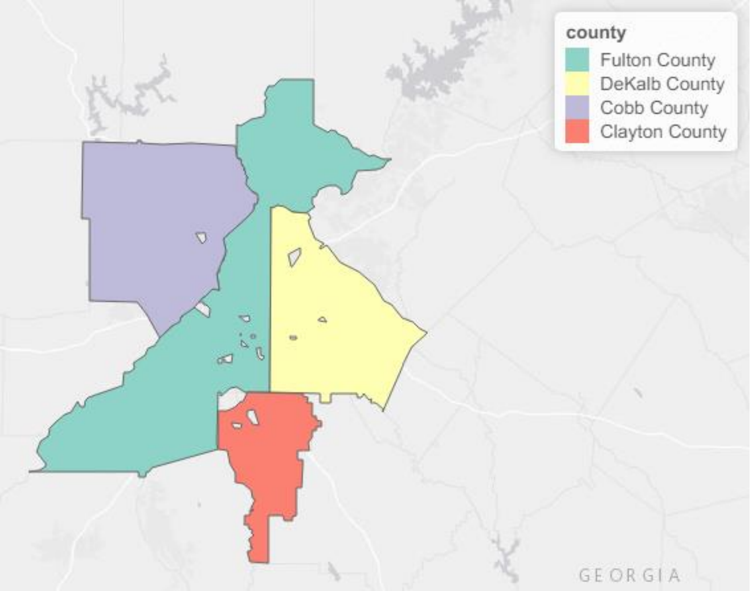
Dissolve using group_by & summarise



```
Simple feature collection with 501 features and 16 fields
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 636450.8 ymin: 371709.7 xmax: 713243.6 ymax: 464175.8
epsg (SRID):    NA
proj4string:    +proj=tmerc +lat_0=30 +lon_0=-84.16666666666667 +k=0.9999 +x_0=700000 +y_0=0 +el
s
First 10 features:
         GEOID  n  walkscr              tract        county   state   mval   hinc nroom
1  13121000100 27 51.55556     Census Tract 1 Fulton County Georgia 680600 149450   7.4
2  13121000200 20 65.70000     Census Tract 2 Fulton County Georgia 548300 116094   6.7
3  13121000400 11 72.00000     Census Tract 4 Fulton County Georgia 419300  91857   4.8
4  13121000500 26 61.19231     Census Tract 5 Fulton County Georgia 343600  85750   3.9
5  13121000600 12 80.91667     Census Tract 6 Fulton County Georgia 190300  50969   3.4
6  13121000700 13 44.15385     Census Tract 7 Fulton County Georgia 258300  71214   5.5
7  13121001001  6 81.33333 Census Tract 10.01 Fulton County Georgia 284600  80132   3.5
8  13121001100  4 90.50000    Census Tract 11 Fulton County Georgia 293700  99205   3.5
9  13121001201  6 89.16667 Census Tract 12.01 Fulton County Georgia 246800  58603   3.6
10 13121001202  3 91.33333 Census Tract 12.02 Fulton County Georgia 227000  83063   3.0
   unt_sng unt_mlt untstr_    p_singl   p_multi  cbd_dst in_atl                       geometry
1     1680     401    2098 0.80076263 0.1911344 4840.388      1 POLYGON ((681417.9 421943.5...
2     1880     991    2871 0.65482410 0.3451759 4027.183      1 POLYGON ((680783.2 421461.6...
3      283     902    1194 0.23701843 0.7554439 3755.205      1 POLYGON ((679545.4 420877.8...
4     1199    2325    3524 0.34023837 0.6597616 4177.759      1 POLYGON ((678314.2 420537.6...
5      368    2678    3046 0.12081418 0.8791858 3661.335      1 POLYGON ((676981.4 419617.1...
6      351      67     418 0.83971292 0.1602871 3684.300      1 POLYGON ((675397.2 419828.9...
7      136    1027    1163 0.11693895 0.8830610 2470.296      1 POLYGON ((678509 419814.7, ...
8      216    3617    3833 0.05635273 0.9436473 3655.297      1 POLYGON ((679519 419633.4, ...
9      296    2550    2869 0.10317184 0.8888114 2506.818      1 POLYGON ((679945.3 418412.7...
10     142    3328    3470 0.04092219 0.9590778 2470.604      1 POLYGON ((679516.5 419342.9...
```

<- this is an sf object census. One of its variables is county.

Let’s say we want to dissolve it by county and make it look like this ➜



Dissolve using group_by & summarise

To do so, you really need to determine two things:

1. A variable with which polygons will be grouped & merged
2. What to do with all other variables (e.g., mval, hinc, etc.) after merging it.

R tackles these with 2-step functions:

1. group_by() function specifies which variable to group it by.
2. summarise() function defines what to do with all other variables. You can take average of mval for each county, for example. Or you can instead use median rather than average.

Spatial Join

First object

Second object

Spatial Join

```
join1 <- st_join(sf object1, sf object2, join = st_intersects)
join2 <- st_join(sf object2, sf object1, join = st_intersects)
```

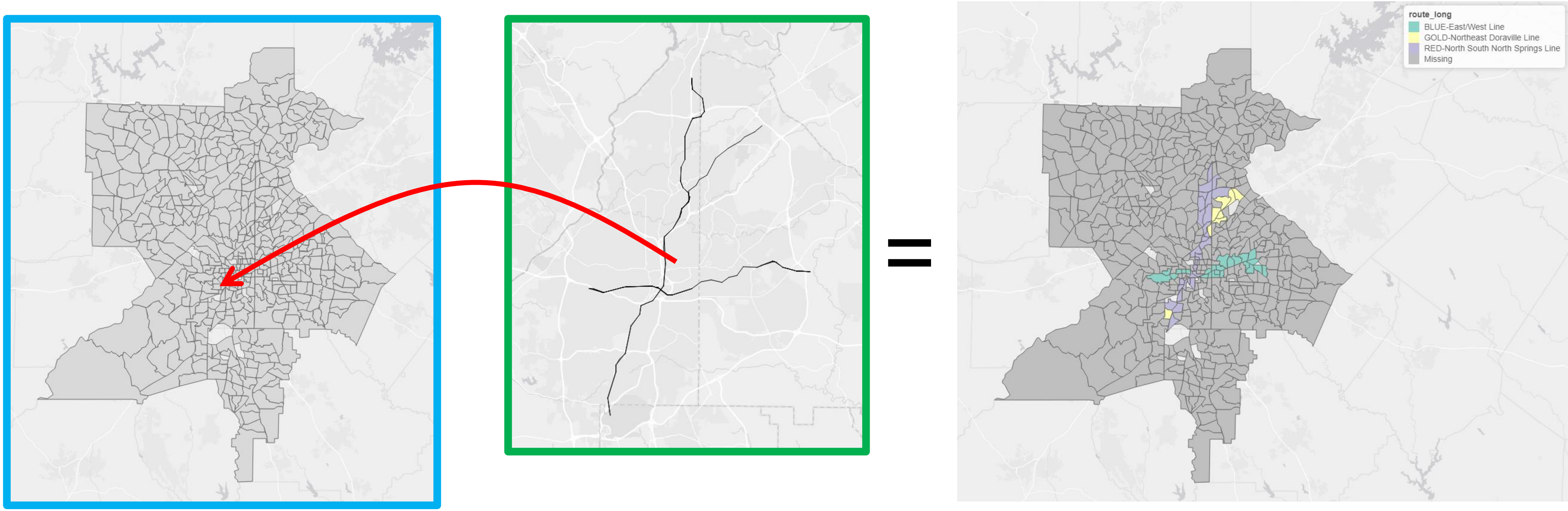
Order matters; join1 and join2 are not same!

The second object is joined to first object.

The output follows the form and type of the first object

Spatial Join

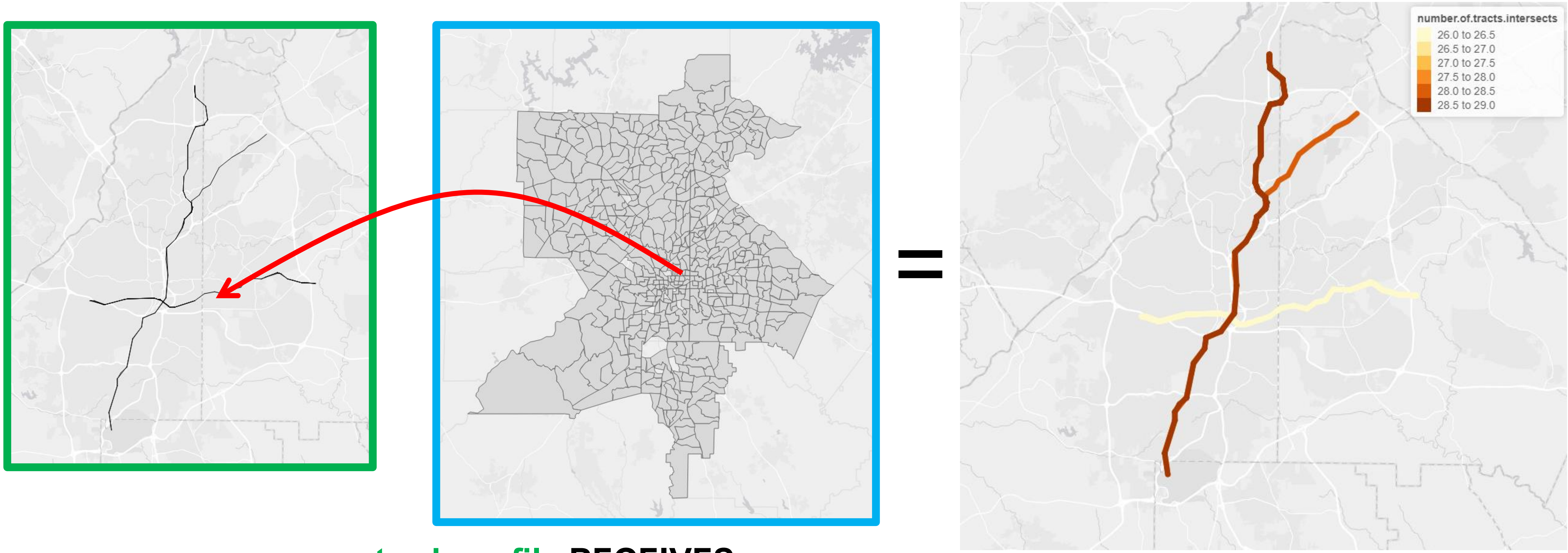
```
st_join(census, marta, join = st_intersects)
```


=
route_long
🟩 BLUE-East/West Line
🟨 GOLD-Northeast Doraville Line
🟪 RED-North South North Springs Line
⬜ Missing

census shapefile RECEIVES
the information in marta shapefile

Spatial Join

st_join(marta, census, join = st_intersects)



marta shapefile RECEIVES
the information in census shapefile

Area / Length / Distance

st_area, st_length, st_distance

The output from these functions is a class of “unit”.

unclass() function converts the output into a normal vector with some additional information

Instead of spending too much time.. Take my word and just wrap st_area, st_length, or st_distance with unclass() function and use it as below.

```
object$area <- unclass(st_area(object))
```

Area / Length / Distance

Area (e.g., Census Tract shapefile):

```
        sf object$area <- unclass(st_area(sf object))
```

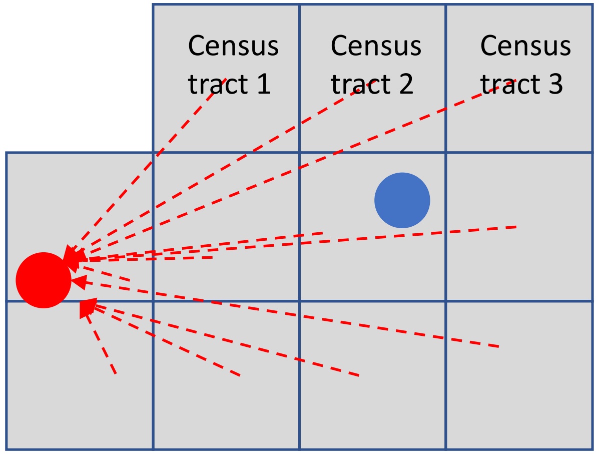
Length (e.g., road shapefile):

```
        sf object$length <- unclass(st_length(sf object))
```

Area / Length / Distance

Distance is a bit different

unclass(st_distance(sf object1, sf object2)) **= a matrix**

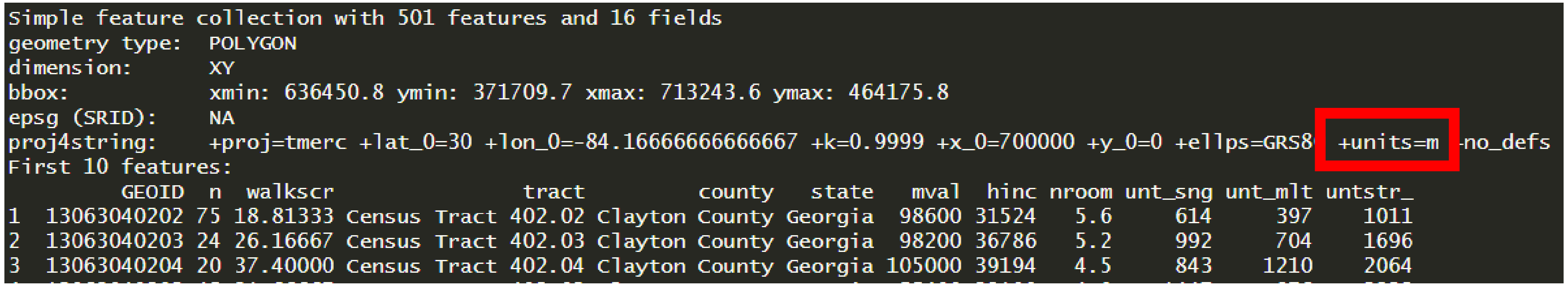


	Distance to RED DOT	Distance to BLUE DOT
Census tract 1	1238	2345
Census tract 2	1256	2346
Census tract 3	356	345
Census tract 4	23	3663
Census tract 5	356	3677
Census tract 6	2856	3755
Census tract 7	2907	3644
Census tract 8	3358	4566
Census tract 9	3208	325
Census tract 10	4985	36
Census tract 11	4850	345
...	...	...

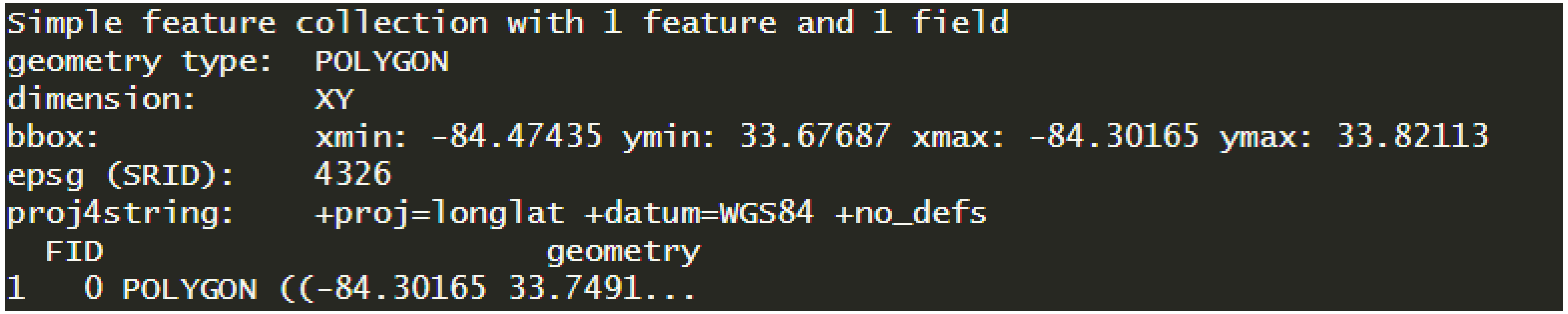
Distance will be between the closest part (not the centroid) of each census tract to the dots

Area / Length / Distance

The measurements uses the unit set by the CRS.



```
Simple feature collection with 501 features and 16 fields
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 636450.8 ymin: 371709.7 xmax: 713243.6 ymax: 464175.8
epsg (SRID):    NA
proj4string:    +proj=tmerc +lat_0=30 +lon_0=-84.16666666666667 +k=0.9999 +x_0=700000 +y_0=0 +ellps=GRS8  +units=m -no_defs
First 10 features:
        GEOID  n  walkscr              tract         county   state   mval  hinc nroom unt_sng unt_mlt untstr_
1  13063040202 75 18.81333 Census Tract 402.02 Clayton County Georgia  98600 31524   5.6     614     397    1011
2  13063040203 24 26.16667 Census Tract 402.03 Clayton County Georgia  98200 36786   5.2     992     704    1696
3  13063040204 20 37.40000 Census Tract 402.04 Clayton County Georgia 105000 39194   4.5     843    1210    2064
```



```
Simple feature collection with 1 feature and 1 field
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: -84.47435 ymin: 33.67687 xmax: -84.30165 ymax: 33.82113
epsg (SRID):    4326
proj4string:    +proj=longlat +datum=WGS84 +no_defs
  FID                       geometry
1   0 POLYGON ((-84.30165 33.7491...
```

<- This sf object does not show what unit it uses. That is because this is not a projected coordinate system, and therefore it uses degree as the unit.

For more info:
https://en.wikipedia.org/wiki/Geographic_coordinate_system#Length_of_a_degree

Open the following link in an internet browser:
http://rpubs.com/bonwoodesign/CP6025_sf_2

Start reading through the document and
try to replicate each step in your R-Studio

Let me know if you have any questions!