Lab 6

Selenea Gibson, (Sgibson2@umbc.edu (mailto:Sgibson2@umbc.edu))

03/7/23

Write 3 variables in a hexagon shapefile then produce the maps in qgis. See if you want to do it in a different format to make all 3 maps be put on one layout in qgis.

Project Overview

In this document, we're going to compute the Moran's I (https://www.statisticshowto.com/morans-i/) for a set of variables that we downloaded from the ACS. There will be exercises on the bottom of this rmarkdown that will be created into maps using `QGIS`.

If the p-value is significant of the Moran's I, then we interpret the value of the Moran's I statistic as follows: For every census tract it is going to see if that value relates to another value. - **-1** is perfect clustering of dissimilar values (you can also think of this as perfect dispersion). - **0** is no autocorrelation (perfect randomness.) - **+1** indicates perfect clustering of similar values (it's the opposite of dispersion).

Setup

See also mapping with TidyCensus (https://walker-data.com/tidycensus/articles/spatial-data.html).

```
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.2.2
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
## Warning: package 'tibble' was built under R version 4.2.1
```

```
## Warning: package 'tidyr' was built under R version 4.2.2
```

```
## Warning: package 'readr' was built under R version 4.2.2
```

```
## Warning: package 'purrr' was built under R version 4.2.2
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
## Warning: package 'stringr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

```
## Warning: package 'lubridate' was built under R version 4.2.2
```

```
library(tigris)
```

```
## Warning: package 'tigris' was built under R version 4.2.2
```

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 4.2.2
```

```
library(dplyr)
library(ggplot2)
library(spdep)
```

```
## Warning: package 'spdep' was built under R version 4.2.2
```

```
## Warning: package 'spData' was built under R version 4.2.2
```

```
library(tmap)
library(gridExtra)
library(maptools)
```

```
## Warning: package 'maptools' was built under R version 4.2.2
```

```
## Warning: package 'sp' was built under R version 4.2.2
```

```
options(tigris_class = "sf") # Make sure we're using sf for tigris
options(tigris_use_cache = TRUE) # Make sure we cache
#census_api_key("YOURAPIKEYHERE", install = TRUE)
```

Baltimore Data

Download data for Baltimore on housing, population, and income. See the Social Explorer Data Dictionary (https://www.socialexplorer.com/data/ACS2021_5yr/metadata/?ds=ACS21_5yr) for more information about these variables.

```
#This gets the 2021 ACS 5-Year population, race+eth, housing units, and median household income
from the ACS
bltmr_tract_20 <- get_acs(geography = "tract",
                          variables = c("pop" = "B03002_001", # Total
                                        "pop_nhwhite" = "B03002_003", # NH White
                                        "pop_nhblack" = "B03002_004", # NH Black
                                        "pop_nhamind" = "B03002_005", # NH Am Ind
                                        "pop_nhasian" = "B03002_006", # NH Asian
                                        "pop_nhhwnpi" = "B03002_007", # NH Hawaiian/PI
                                        "pop_nhother" = "B03002_008", # One Other
                                        "pop_nhtwomr" = "B03002_009", # Two+
                                        "pop_hispltx" = "B03002_012", # Hispanic/Latinx
                                        "hu_total"  = "B25001_001", # Housing Units
                                        "hu_totocc" = "B25003_001", # Housing Units - Occ
                                        "hu_totown" = "B25003_002", # Housing Units - Owner Oc
c,
                                        "hu_totrnt" = "B25003_003", # Housing Units - Renter Oc
c,
                                        "mhhi" = "B19013_001"),

                          year = 2021,
                          survey = "acs5",
                          state = c(24),
                          county = c(510),
                          geometry = TRUE,
                          output = "wide")
```

```
## Getting data from the 2017-2021 5-year ACS
```

```
#transform the data to crs 3857
bltmr_tract_20 <- st_transform(bltmr_tract_20, 3857)
```

Computing Calculations

With these commands, we problematically, but conveniently, reduce the **race+eth** into 5 categories:

- NH White
- NH Black
- Hisp/Latx
- NH Asian
- NH Multi/Other

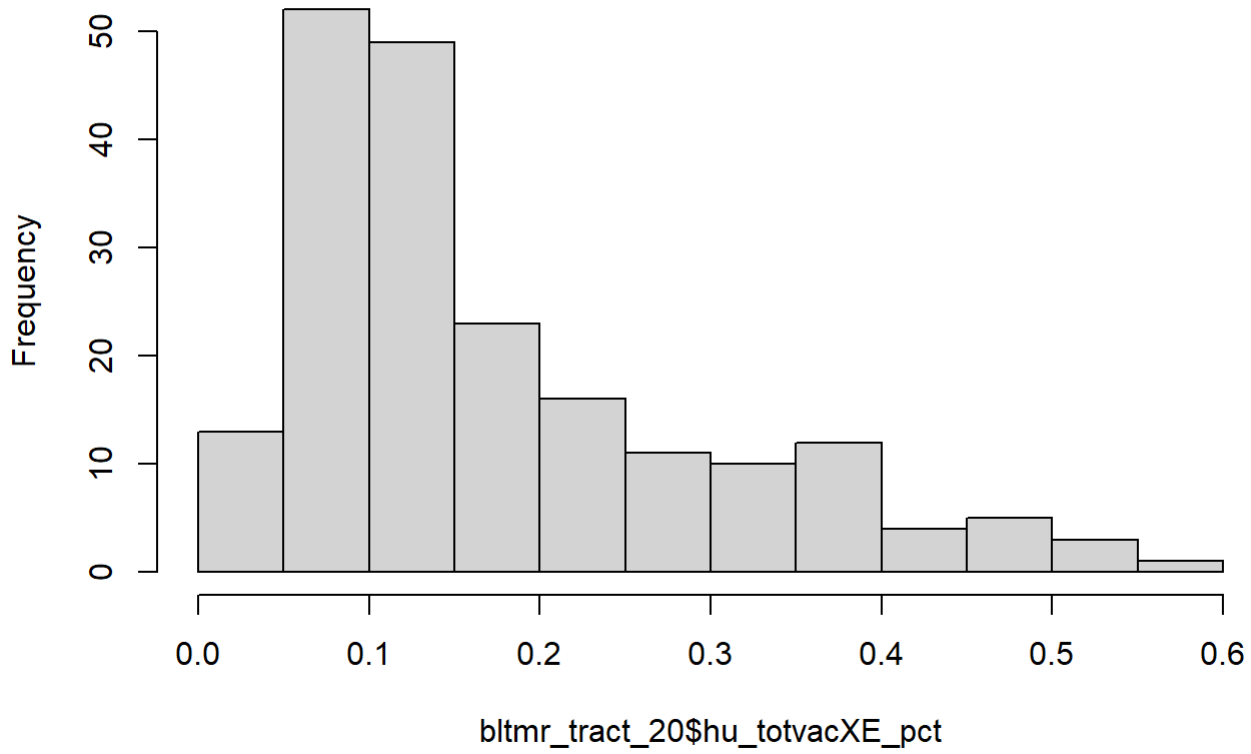
The last set of command also computes the number of occupied houses by the total number of housing units to calculate vacancy rate. For the calculations we are making a new variable that has an “X” to indicate that it is a new variable from the calculations that we have completed.

```
#Computes the NH Asian Population taking pop_nhasian plus nhhwnpi
bltmr_tract_20$pop_nhasianXE <-
  bltmr_tract_20$pop_nhasianE + bltmr_tract_20$pop_nhhwnpiE

#Computes the NH "Other" Population taking nhamind plus nhother plus nhtwomr
bltmr_tract_20$pop_nhotherXE <-
  bltmr_tract_20$pop_nhamindE + bltmr_tract_20$pop_nhotherE + bltmr_tract_20$pop_nhtwomrE

#Computes vacant housing (need to verify these are the correct values) subtracting the hu_total
from hu_totocc
bltmr_tract_20$hu_totvacXE <- bltmr_tract_20$hu_totalE - bltmr_tract_20$hu_totoccE

#computes the housing total vacant percent by dividing hu_totalvacxe by hu_total
bltmr_tract_20$hu_totvacXE_pct <- bltmr_tract_20$hu_totvacXE/ bltmr_tract_20$hu_totalE

#Replace NaNs with 0 because people may not be living in certain tracts an it is missing data no
t na
bltmr_tract_20$hu_totvacXE_pct[is.nan(bltmr_tract_20$hu_totvacXE_pct)] <- 0
bltmr_tract_20$hu_totvacXE_pct <- as.numeric(bltmr_tract_20$hu_totvacXE_pct)
```

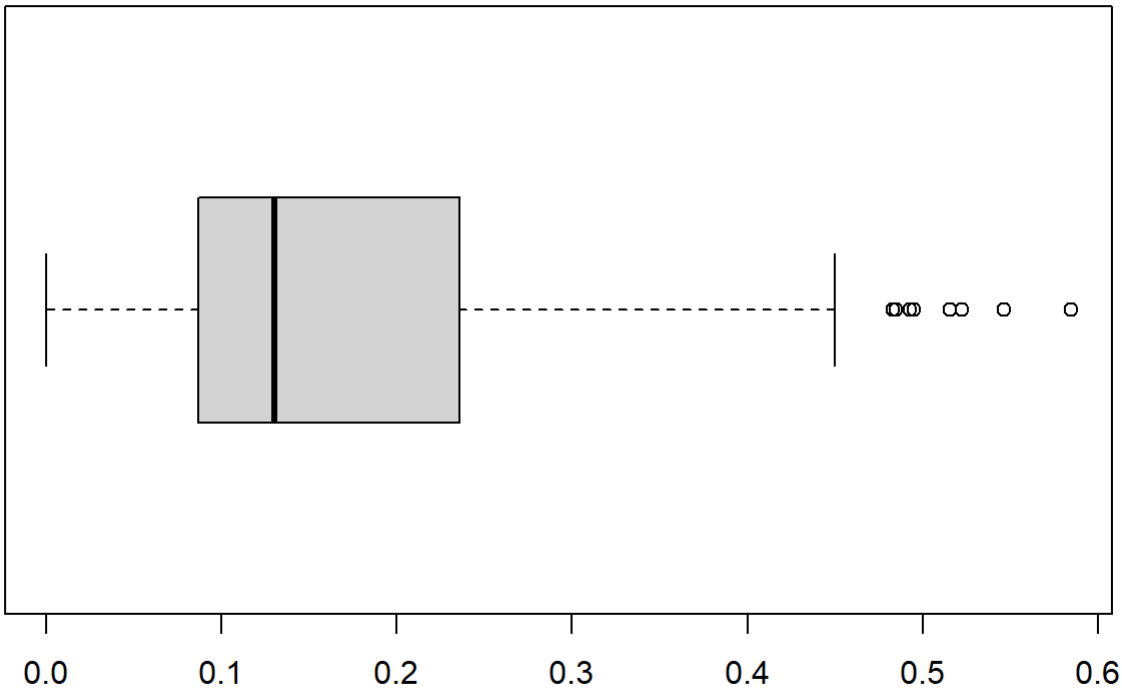
Analyze Vacancy

See this example (https://mgimond.github.io/simple_moransI_example/). Let's just take a look at the distribution of vacancy rates.

```
#creating a histogram of the hu_total_vacXE_pct variable, no main title so it is set to NULL
hist(bltmr_tract_20$hu_totvacXE_pct, main=NULL)
```



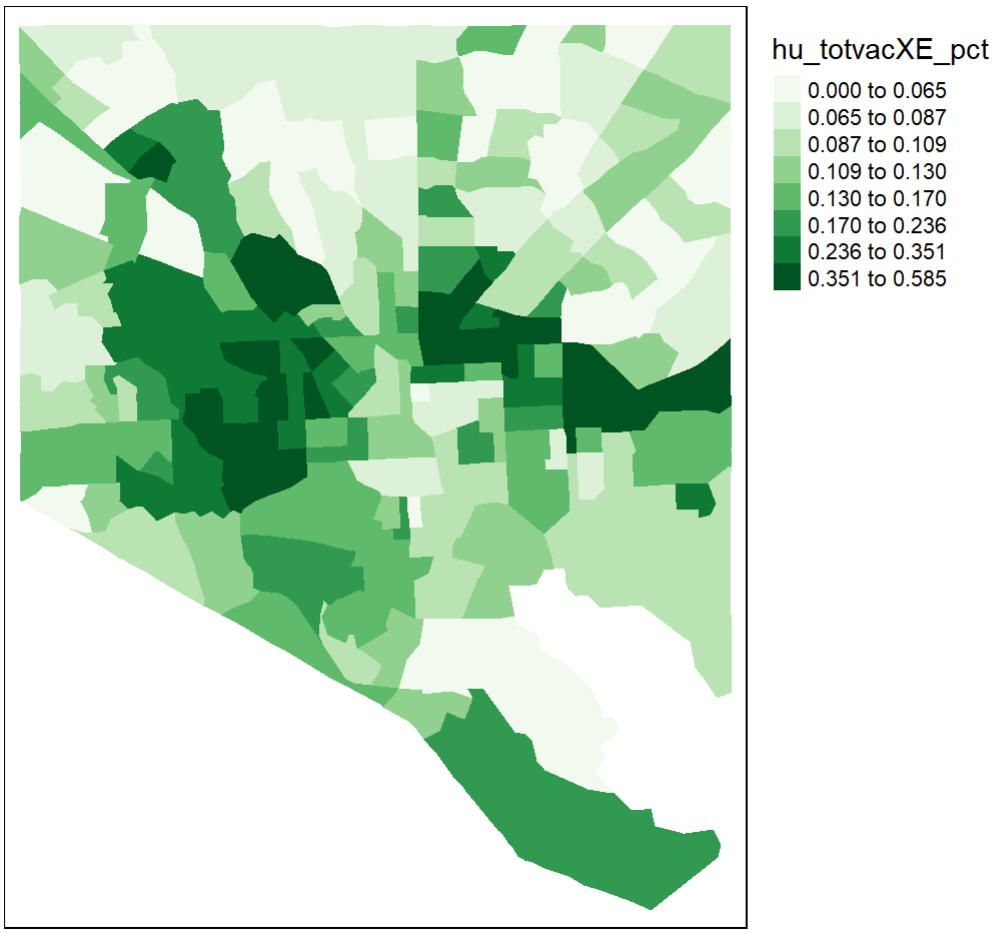

```
#creating a boxplot of the hu_total_vacXE_pct variable, showing a mean with the horizontal line
= TRUE
boxplot(bltmr_tract_20$hu_totvacXE_pct, horizontal = TRUE)
```



Tmap

Quick map using `tmap` library.

```
#applying the column to be our hu_totvacXE_pct, the style being quantile with 8 classifications,
and greens for our colors.
tm_shape(bltmr_tract_20) + tm_fill(col="hu_totvacXE_pct", style="quantile", n=8, palette="Green
s") +
              tm_legend(outside=TRUE)
```



Matrix

Create a matrix of neighboring (adjacent) polygons for each tract. The `poly2nb` function creates a list of adjacent polygons (https://www.rdocumentation.org/packages/spdep/versions/1.2-7/topics/poly2nb) for each polygon. Then `nb2listw` puts this list in a spatial weights format (https://www.rdocumentation.org/packages/spdep/versions/1.2-7/topics/nb2listw) that we need for the Moran's I.

```
#creates neighbor/adjacency list
bltmr_tract_nb  <- poly2nb(bltmr_tract_20, queen=TRUE)

#puts in a listw format
bltmr_tract_lw  <- nb2listw(bltmr_tract_nb, style="W", zero.policy=TRUE)
```

Compute Moran's I.

Here I am going to compute the Moran's I using the `moran.test` function. This will be on the vacancy column.

```
#compute Moran's I on vacancy and store in a variable, notice the listw input
bltmr_tract_moran.vac <- moran.test(bltmr_tract_20$hu_totvacXE_pct, bltmr_tract_lw)

#show the Moran's I output
bltmr_tract_moran.vac
```

```
##
##  Moran I test under randomisation
##
## data:  bltmr_tract_20$hu_totvacXE_pct
## weights: bltmr_tract_lw
##
## Moran I statistic standard deviate = 14.501, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.588653172      -0.005050505       0.001676239
```

Same calculation as above but using the `mhhiE` variable instead.

```
#replace NA with 0 for median household income - a better solution might be to remove them befor
e computing the poly2nb or the nb2listw
bltmr_tract_20$mhhiE[is.na(bltmr_tract_20$mhhiE)] <- 0

#compute Moran's I and store in a variable
bltmr_tract_moran.mhhi <- moran.test(bltmr_tract_20$mhhiE, bltmr_tract_lw)

#show the Moran's I output
bltmr_tract_moran.mhhi
```

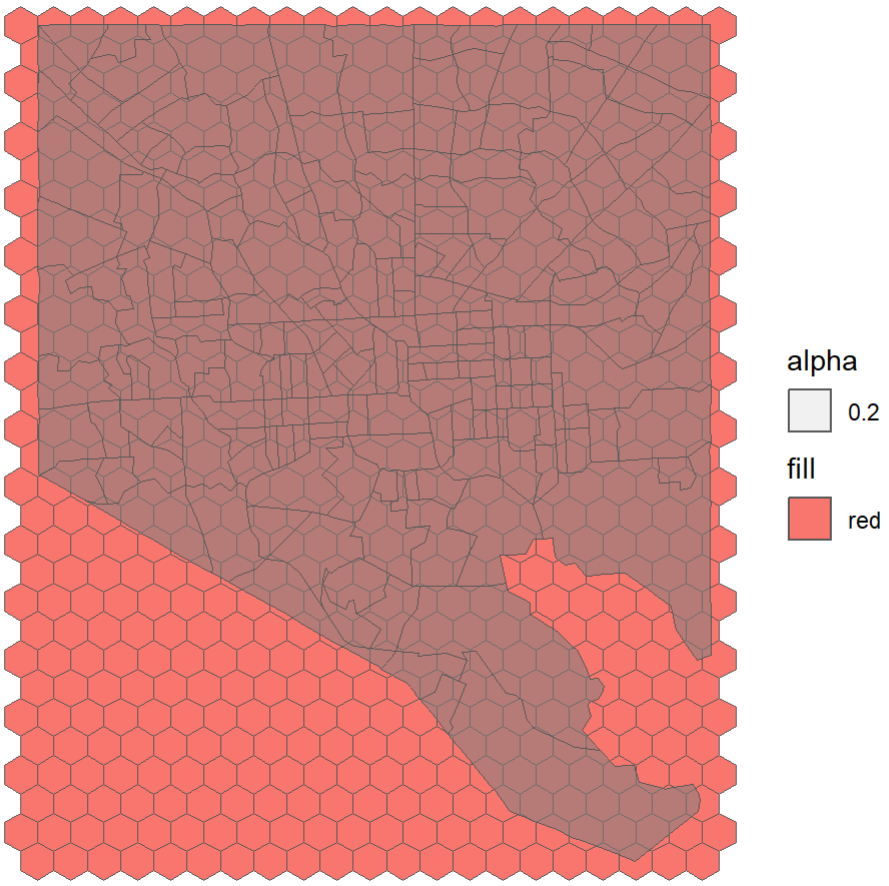
```
##
##  Moran I test under randomisation
##
## data:  bltmr_tract_20$mhhiE
## weights: bltmr_tract_lw
##
## Moran I statistic standard deviate = 12.447, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.500345629      -0.005050505       0.001648545
```

Because the value of the computed Moran's I is above 0, at 0.5003456, -0.0050505, 0.0016485, and the p-value is significant at 7.216181^{-36}, we say that there is less than 1% chance that the phenomenon is not clustered.

Using the data

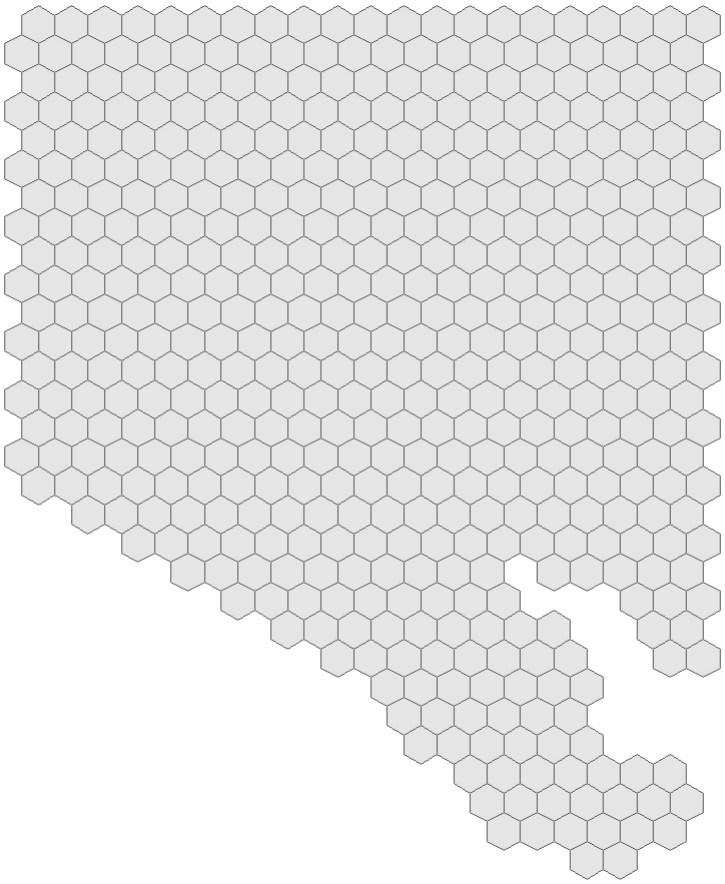
Here we make a grid, note the piping to make it an `sf` object.

```
#3857 is in meters, so this is 1km
bltmr_hex <- st_make_grid(bltmr_tract_20,
  1 * 1000,
  crs = st_crs(bltmr_tract_20),
  what = "polygons",
  square = FALSE) %>%
  st_sf %>% # make an sf object
  st_cast # make polygons


#create an ID for each row
#bltmr_hex$GEOID <- 0 # this forces it to be a number, sets all values to 0
bltmr_hex$GEOID <- seq.int(nrow(bltmr_hex))

#plotting the bltmr_hex variable with a fill of red and plotting our bltmr_tract_20 variable wit
h no fill and a 20% transparency
ggplot() + geom_sf(data=bltmr_hex, aes(fill="red")) +
  geom_sf(data=bltmr_tract_20, aes(fill = NA, alpha = 0.2)) + # 20% transparency
  theme_void()
```



We can spatially subset (filter or intersect) this grid to what overlaps the city (or specifically the tracts of the city).

```
#filtering the bltmr_hex to the tract_20 and seeing where they intersect
bltmr_hex_20 <- st_filter(bltmr_hex, bltmr_tract_20, .predicate=st_intersects)

#plotting the new variable created from the filter
ggplot() + geom_sf(data=bltmr_hex_20, aes()) + theme_void()
```



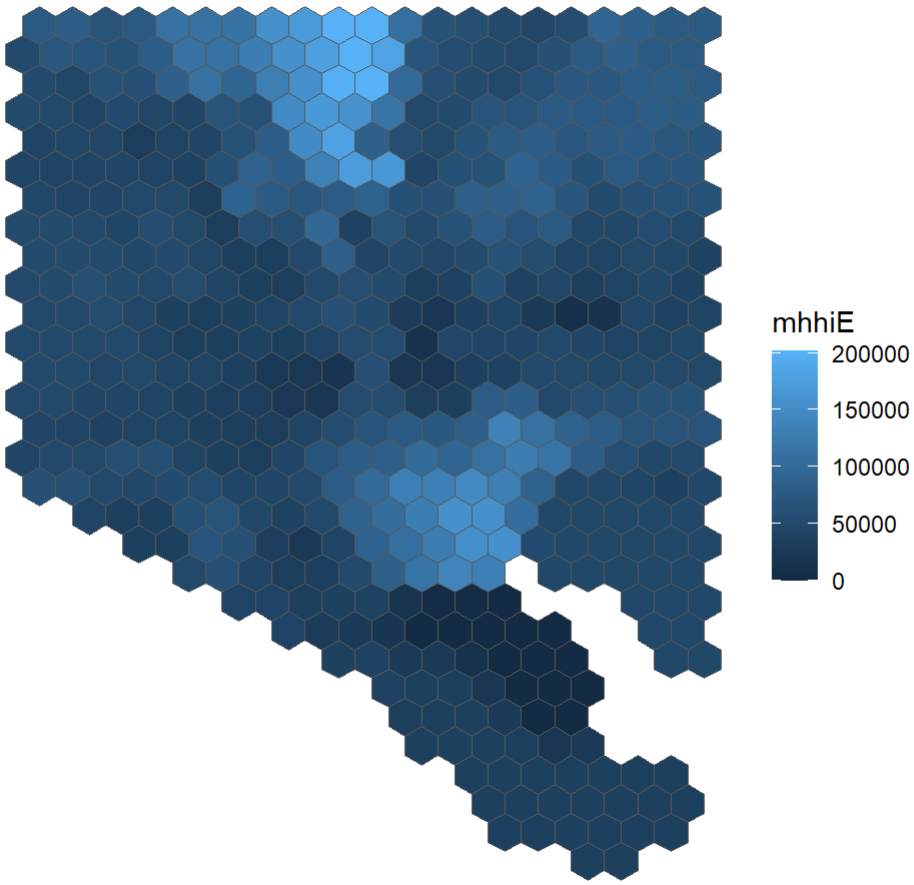
Area weight

Now we're going to compute the area weight median household income into the hexagons. Kyle walker explains the `interpolate_aw` function as "a user computes area-weighted areal interpolation with st_interpolate_aw(), the function prints the following warning: st_interpolate_aw assumes attributes are constant or uniform over areas of x. This assumption that proportionally larger areas also have proportionally more people is often incorrect with respect to the geography of human settlements, and can be a source of error when using this method. An alternative method, population-weighted areal interpolation, can represent an improvement. As opposed to using area-based weights, population-weighted techniques estimate the populations of the intersections between origin and destination from a third dataset, then use those values for interpolation weights (Walker 2022)."

```
# overwriting this variable
#area interpolation means we are taking the tracts and pulling only 1 variable from the tracts t
o the hexagons
bltmr_hex_20.mhhi <- st_interpolate_aw(bltmr_tract_20["mhhiE"], bltmr_hex_20, extensive = FALSE)
```

```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["mhhiE"], bltmr_hex_20,
## extensive = FALSE): st_interpolate_aw assumes attributes are constant or
## uniform over areas of x
```

```
#ploting our area weighted variable with the fill being the mhhiE column
ggplot() +
  geom_sf(data=bltmr_hex_20.mhhi, aes(fill = mhhiE)) +
  theme_void()
```



Write out

Writing out the results to our results folder.

```
#writing out the plot to a geojson in our results folder
st_write(bltmr_hex_20.mhhi, dsn = "D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr_h
ex_20.mhhi.geojson", layer = "bltmr_hex_20.mhhi.geojson", append = FALSE)
```

```
## Warning in CPL_write_ogr(obj, dsn, layer, driver,
## as.character(dataset_options), : GDAL Error 6: DeleteLayer() not supported by
## this dataset.
```

```
## Deleting layer not supported by driver `GeoJSON'
## Deleting layer `bltmr_hex_20.mhhi.geojson' failed
## Writing layer `bltmr_hex_20.mhhi.geojson' to data source
##   `D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr_hex_20.mhhi.geojson' using dri
ver `GeoJSON'
## Updating existing layer bltmr_hex_20.mhhi.geojson
## Writing 483 features with 1 fields and geometry type Polygon.
```

Exercises

Here I will compute numerous exercises for the following questions.

1. Compute the area weight for a) percent black and b) vacancy rates using hexagons.

```
#need this to be the percent black
bltmr_hex_20.pctblk <- st_interpolate_aw(bltmr_tract_20["pop_nhblackE"], bltmr_hex_20, extensive
= FALSE)
```

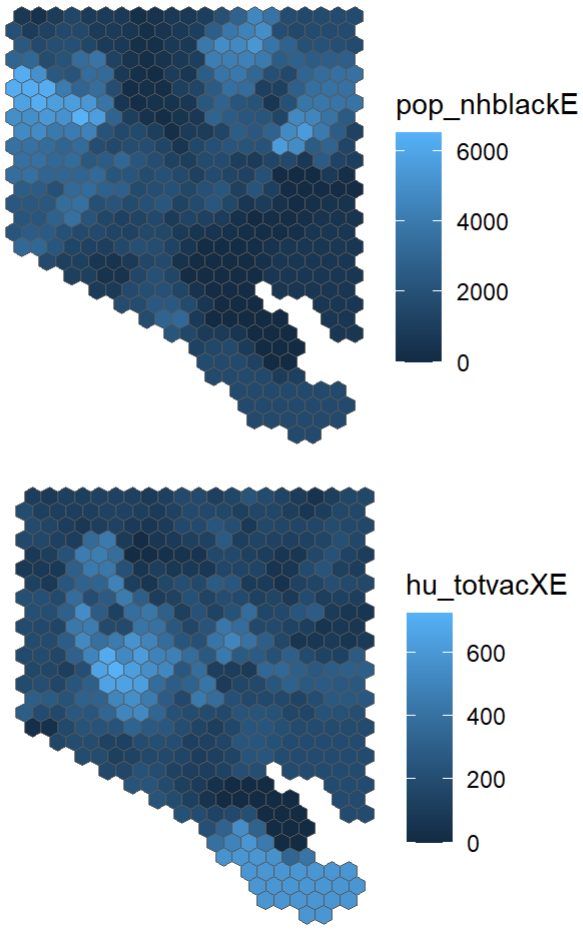
```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["pop_nhblackE"], bltmr_hex_20, :
## st_interpolate_aw assumes attributes are constant or uniform over areas of x
```

```
#need another one for vacancy rates
bltmr_hex_20.vacancy <- st_interpolate_aw(bltmr_tract_20["hu_totvacXE"], bltmr_hex_20, extensive
= FALSE)
```

```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["hu_totvacXE"], bltmr_hex_20, :
## st_interpolate_aw assumes attributes are constant or uniform over areas of x
```

2. Creating maps of both variables.

```
#ploting our bltmr_hex_20.pctblk area weighted with the fill being pop_nhblackE .
bltmr_hex_20.pctblk_plot <- ggplot() +
  geom_sf(data=bltmr_hex_20.pctblk, aes(fill = pop_nhblackE)) +
  theme_void()

##ploting our bltmr_hex_20.vacancy area weighted with the fill being hu_totvacXE .
bltmr_hex_20.vacancy_plot <- ggplot()+
   geom_sf(data=bltmr_hex_20.vacancy, aes(fill = hu_totvacXE)) +
   theme_void()

##ploting our area weighted variables side by side with the new created variables.
grid.arrange(bltmr_hex_20.pctblk_plot,bltmr_hex_20.vacancy_plot)
```



3. Compute the Moran's I of all three variables (percent black, vacancy rates, and median household income) at the hexagon level. Write a couple of sentences interpreting each result.

We do not have a percent black calculation in this dataset. I went ahead and made a new column that takes the black population/total population.

```
#pct_black calculation
bltmr_tract_20$pct_blk <- (bltmr_tract_20$pop_nhblackE/bltmr_tract_20$popE)

#this is used to get rid of any na values in the dataset
bltmr_tract_20[is.na(bltmr_tract_20)] <- 0
```

Finding the Moran's I at the hexagon level for the pct_blk.

```
#this is for making the polygons and then a list w for the spatial weights
bltmr_hex_nb  <- poly2nb(bltmr_tract_20, queen=TRUE)
bltmr_hex_lw  <- nb2listw(bltmr_hex_nb, style="W", zero.policy=TRUE)

#making a morans I for the pct_black column
bltmr_hex_moran.pct_blk <- moran.test(bltmr_tract_20$pct_blk, bltmr_hex_lw)

#looking at the morans I
bltmr_hex_moran.pct_blk
```

```
##
##  Moran I test under randomisation
##
## data:  bltmr_tract_20$pct_blk
## weights: bltmr_hex_lw
##
## Moran I statistic standard deviate = 15.74, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.642256611      -0.005050505       0.001691258
```

Finding the Moran's I at the hexagon level for the vacancy rates.

```
#this is for making the polygons and then a list w for the spatial weights
bltmr_hex_nb  <- poly2nb(bltmr_tract_20, queen=TRUE)
bltmr_hex_lw  <- nb2listw(bltmr_hex_nb, style="W", zero.policy=TRUE)

#making a morans I for the hu_totvacXE_pct column
bltmr_hex_moran.vac_rates <- moran.test(bltmr_tract_20$hu_totvacXE_pct, bltmr_hex_lw)

#Looking at the morans I
bltmr_hex_moran.vac_rates
```

```
##
##  Moran I test under randomisation
##
## data:  bltmr_tract_20$hu_totvacXE_pct
## weights: bltmr_hex_lw
##
## Moran I statistic standard deviate = 14.501, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.588653172      -0.005050505       0.001676239
```

Finding the Moran's I at the hexagon level for the median household income.

```
#this is for making the polygons and then a list w for the spatial weights
bltmr_hex_nb  <- poly2nb(bltmr_tract_20, queen=TRUE)
bltmr_hex_lw  <- nb2listw(bltmr_hex_nb, style="W", zero.policy=TRUE)

#making a morans I for the mhhiE column
bltmr_hex_moran.medhhi <- moran.test(bltmr_tract_20$mhhiE, bltmr_hex_lw)

#Looking at the morans I
bltmr_hex_moran.medhhi
```

```
##
##  Moran I test under randomisation
##
## data:  bltmr_tract_20$mhhiE
## weights: bltmr_hex_lw
##
## Moran I statistic standard deviate = 12.447, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.500345629      -0.005050505       0.001648545
```

Morans I write-up

1. For the percent Black Moran's I we have a statistic of .64 and a p-value that is significant which indicates a less than 1% chance that the phenomenon is not clustered.

2. For the vacancy rates Moran's I we have a statistic of .58 and a p-value that is significant which indicates a less than 1% chance that the phenomenon is not clustered.

3. For the median household income Moran's I we have a statistic of .50 and a p-value that is significant which indicates a less than 1% chance that the phenomenon is not clustered.

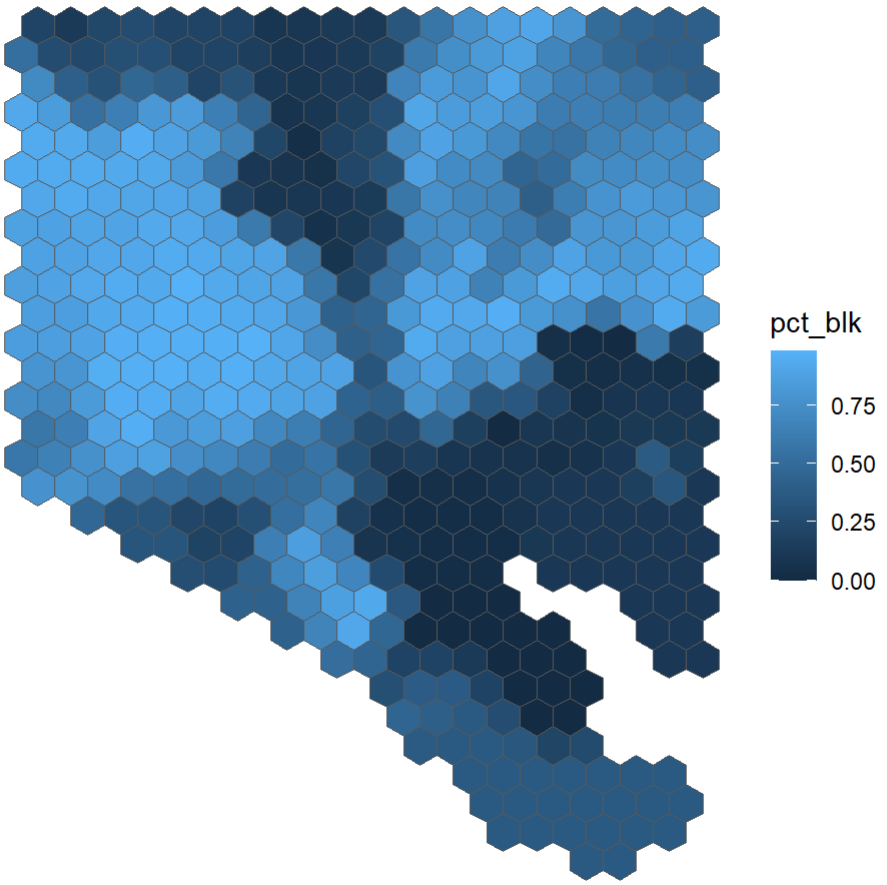
Area interpolation part 2

Here I am going to do the area interpolation like before, but this time it will be with the new variables that I created. The reason why we are doing this again is because we can not write out our morans I test variable to a file. What we need to do is run the `st_interpolate_aw` command again for it to be uniform with the Baltimore city tracts layer.

```
# overwriting this variable
#area interpolation means we are taking the tracts and pulling only 1 variable from the tracts t
o the hexagons
bltmr_hex_20.pctblk <- st_interpolate_aw(bltmr_tract_20["pct_blk"], bltmr_hex_20, extensive = FA
LSE)
```

```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["pct_blk"], bltmr_hex_20, :
## st_interpolate_aw assumes attributes are constant or uniform over areas of x
```

```
#ploting our area weighted variable with the fill being the pct_blk column
ggplot() +
  geom_sf(data=bltmr_hex_20.pctblk, aes(fill = pct_blk)) +
  theme_void()
```

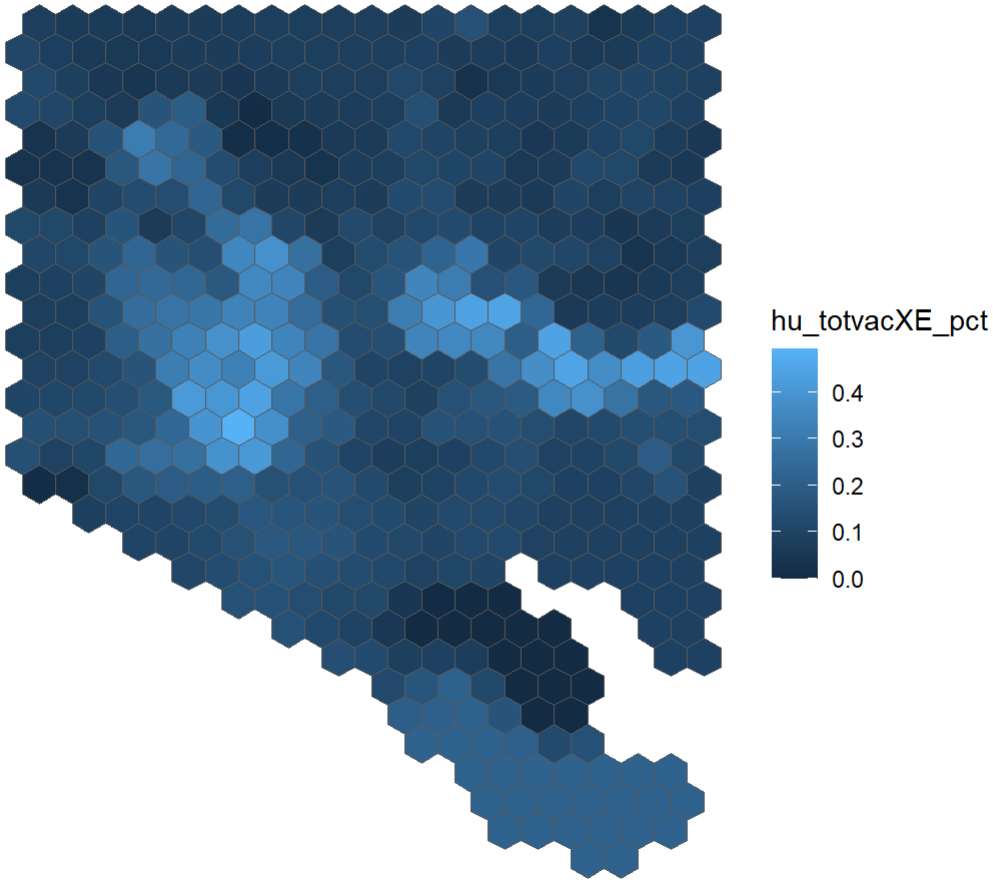


This is for the vacancy rates percents area interpolation.

```
# overwriting this variable
#area interpolation means we are taking the tracts and pulling only 1 variable from the tracts t
o the hexagons
bltmr_hex_20.vacrate <- st_interpolate_aw(bltmr_tract_20["hu_totvacXE_pct"], bltmr_hex_20, exten
sive = FALSE)
```

```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["hu_totvacXE_pct"],
## bltmr_hex_20, : st_interpolate_aw assumes attributes are constant or uniform
## over areas of x
```

```
#ploting our area weighted variable with the fill being the hu_totvacXE_pct column
ggplot() +
  geom_sf(data=bltmr_hex_20.vacrate, aes(fill = hu_totvacXE_pct)) +
  theme_void()
```

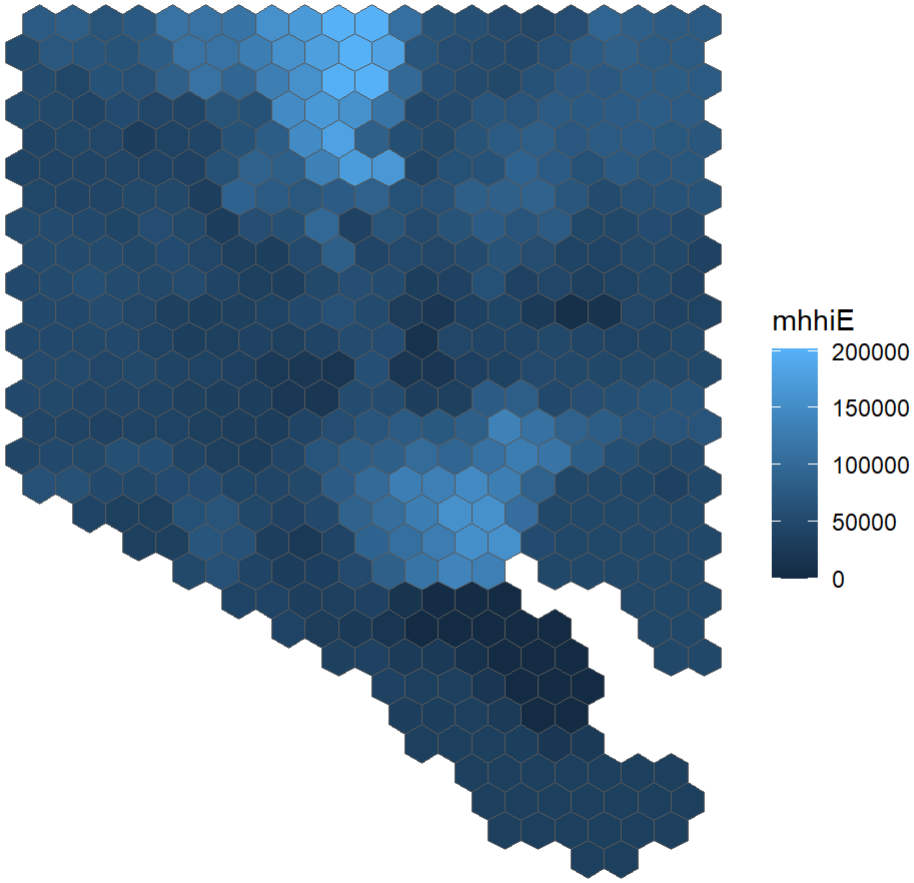


This is for the median household income area interpolation.

```
# overwriting this variable
#area interpolation means we are taking the tracts and pulling only 1 variable from the tracts t
o the hexagons
bltmr_hex_20.medinc <- st_interpolate_aw(bltmr_tract_20["mhhiE"], bltmr_hex_20, extensive = FALS
E)
```

```
## Warning in st_interpolate_aw.sf(bltmr_tract_20["mhhiE"], bltmr_hex_20,
## extensive = FALSE): st_interpolate_aw assumes attributes are constant or
## uniform over areas of x
```

```
#ploting our area weighted variable with the fill being the mhhiE column
ggplot() +
  geom_sf(data=bltmr_hex_20.medinc, aes(fill = mhhiE)) +
  theme_void()
```



4. Export the three variables as ".geojson" to make things easier to import into QGIS.

```
#writing out to geojson
st_write(bltmr_hex_20.pctblk, dsn = "D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr
_hex_20.pctblk.geojson", layer = "bltmr_hex_20.pctblk.geojson", append = FALSE)
```

```
## Warning in CPL_write_ogr(obj, dsn, layer, driver,
## as.character(dataset_options), : GDAL Error 6: DeleteLayer() not supported by
## this dataset.
```

```
## Deleting layer not supported by driver `GeoJSON'
## Deleting layer `bltmr_hex_20.pctblk.geojson' failed
## Writing layer `bltmr_hex_20.pctblk.geojson' to data source
##   `D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr_hex_20.pctblk.geojson' using d
river `GeoJSON'
## Updating existing layer bltmr_hex_20.pctblk.geojson
## Writing 483 features with 1 fields and geometry type Polygon.
```

```
st_write(bltmr_hex_20.vacrate, dsn = "D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltm
r_hex_20.vacrate.geojson", layer = "bltmr_hex_20.vacrate.geojson", append = FALSE)
```

```
## Warning in CPL_write_ogr(obj, dsn, layer, driver,
## as.character(dataset_options), : GDAL Error 6: DeleteLayer() not supported by
## this dataset.
```

```
## Deleting layer not supported by driver `GeoJSON'
## Deleting layer `bltmr_hex_20.vacrate.geojson' failed
## Writing layer `bltmr_hex_20.vacrate.geojson' to data source
##   `D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr_hex_20.vacrate.geojson' using
driver `GeoJSON'
## Updating existing layer bltmr_hex_20.vacrate.geojson
## Writing 483 features with 1 fields and geometry type Polygon.
```

```
st_write(bltmr_hex_20.medinc, dsn = "D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr
_hex_20.medinc.geojson", layer = "bltmr_hex_20.medinc.geojson", append = FALSE)
```

```
## Warning in CPL_write_ogr(obj, dsn, layer, driver,
## as.character(dataset_options), : GDAL Error 6: DeleteLayer() not supported by
## this dataset.
```

```
## Deleting layer not supported by driver `GeoJSON'
## Deleting layer `bltmr_hex_20.medinc.geojson' failed
## Writing layer `bltmr_hex_20.medinc.geojson' to data source
##   `D:/UMBC/Grad_School/Spring_2023/GES_687/Lab_6/results/bltmr_hex_20.medinc.geojson' using d
river `GeoJSON'
## Updating existing layer bltmr_hex_20.medinc.geojson
## Writing 483 features with 1 fields and geometry type Polygon.
```

5. Make set of polished maps in QGIS using the layout manager.

Note: this will be completed in the QGIS application therefore this rmd is completed.

Bibliography

Walker, Kyle E. 2020. Chapter 7 Spatial Analysis with US Census Data | Analyzing US Census Data. Accessed March 8, 2023. Walker Data (https://walker-data.com/census-r/spatial-analysis-with-us-census-data.html).