DSCI 210: US Census Data MAPS in the tidyverse

David Gerberry

05 September, 2023

Introduction

Again, we are using the same resources. For a general overview of what we will refer to generically as "Census Data" see the information from the Demographics Research Group at the University of Virginia at

- https://demographics.coopercenter.org/guide-to-publicly-available-demographic-data/

and the book, *Data Wrangling*, by Altman, Behrman, and Wickham which available free at

- https://dcl-wrangle.stanford.edu/census.html

So you are aware, to build this lesson on "tidycensus" I am closely following the description at

- https://walker-data.com/tidycensus/.

It seems this Walker person has a whole online book on doing analyses with Census data in R.

- https://walker-data.com/census-r/

Getting started

To get started working with tidycensus, users should load the package along with the tidyverse package, and set their Census API key. A key can be obtained from http://api.census.gov/data/key_signup.html. The Census key included below is MINE... get your own!! We'll talk more about API's later in the semester. You'll also hear about them in BAIS 396.

```
knitr::opts_chunk$set(
    echo = TRUE,
    message = FALSE,
    warning = FALSE
)
library(tidycensus)
library(tidyverse)
library(sf)
library(tigris)

census_api_key("fa67b1dbacf4fbbb1b14c875f34437c6cbdaa694")
```

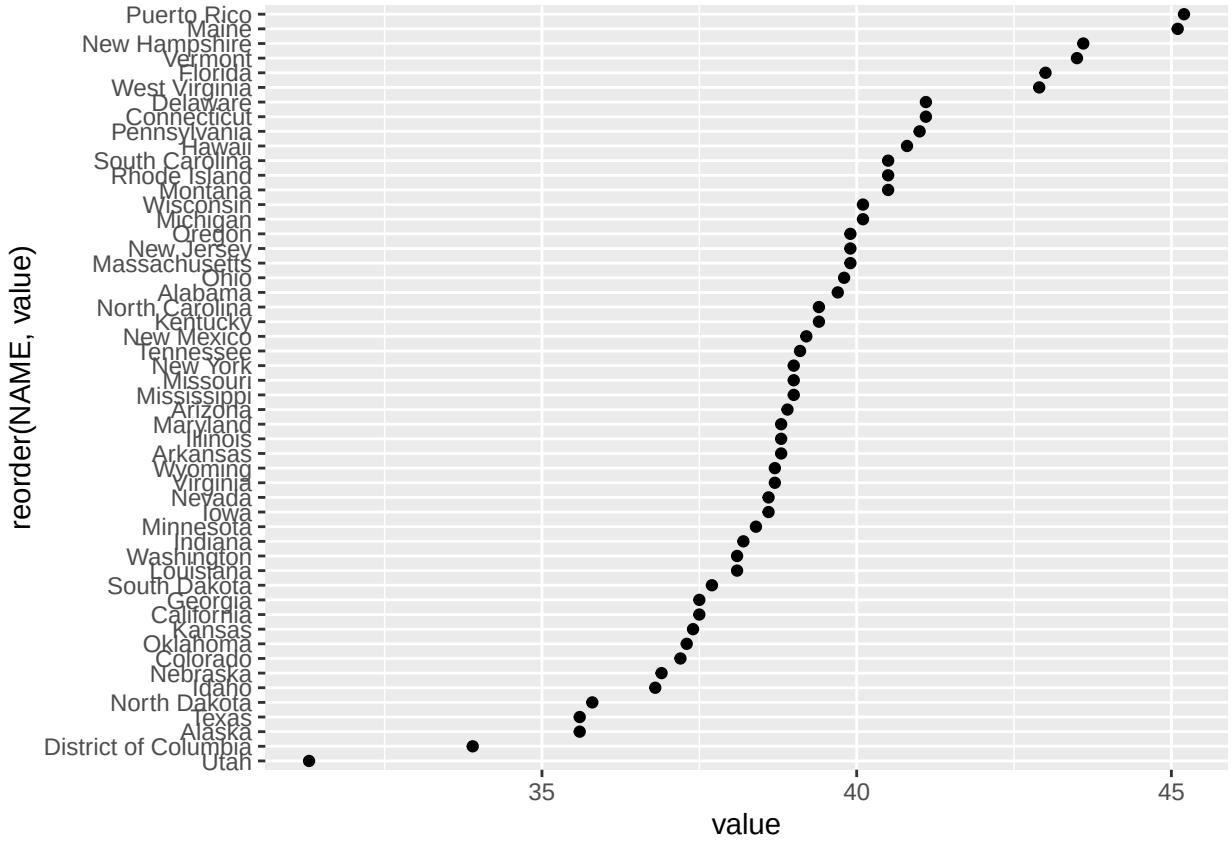
Median age by state

So let's demonstrate how cool and easy this can be. We'll look in the 2020 US Census data and grab the median age for each state. It's important to note that we haven't downloaded the entire US Census data set. That would be insane.
We are using the API to go to the US Census servers and grab exactly the data that we want to use.

```
age20 <- get_decennial(geography = "state",
                       variables = "P13_001N",
                       year = 2020,
                       sumfile = "dhc")
```

We have the data now, so what do we do with it? We plot it somehow to see if there's anything interesting in there. How? Use ggplot().

```
age20 %>%
  ggplot(aes(x = value, y = reorder(NAME, value))) +
  geom_point()
```



Searching for variables

The first question that should have come to mind, "Where in the world did"P13_001N" come from?" The answer to this question kind of sucks. Here is what https://walker-data.com/tidycensus/ tells us about about this.

> Getting variables from the Census or ACS requires knowing the variable ID - and there are thousands of these IDs across the different Census files. To rapidly search for variables, use the load_variables() function. The function takes two required arguments: the year of the Census or endyear of the ACS sample, and the dataset name, which varies in availability by year. For the decennial Census, possible dataset choices include "pl" for the redistricting files; "dhc" for the Demographic and Housing Characteristics file and "dp" for the Demographic Profile (2020 only), and "sf1" or "sf2" (2000 and 2010) and "sf3" or "sf4" (2000 only) for the various summary files. Special island area summary files are available with "as", "mp", "gu", or "vi".
>
> For the ACS, use either "acs1" or "acs5" for the ACS detailed tables, and append /profile for the

Data Profile and /subject for the Subject Tables. To browse these variables, assign the result of this function to a variable and use the View function in RStudio. An optional argument cache = TRUE will cache the dataset on your computer for future use.

```
dhc.vars <- load_variables(2020, "dhc", cache = TRUE)
```

We can then view "dhc.vars" like we view any table in R with View(dhc.vars) which isn't super encouraging because it's a table that has over 9,000 rows. At this point, we take advantage of the search bar that is available in the Viewer to search for the phrase "median age." Doing so, we find that the code "P13_001N'' gives us the median age for both sexes.

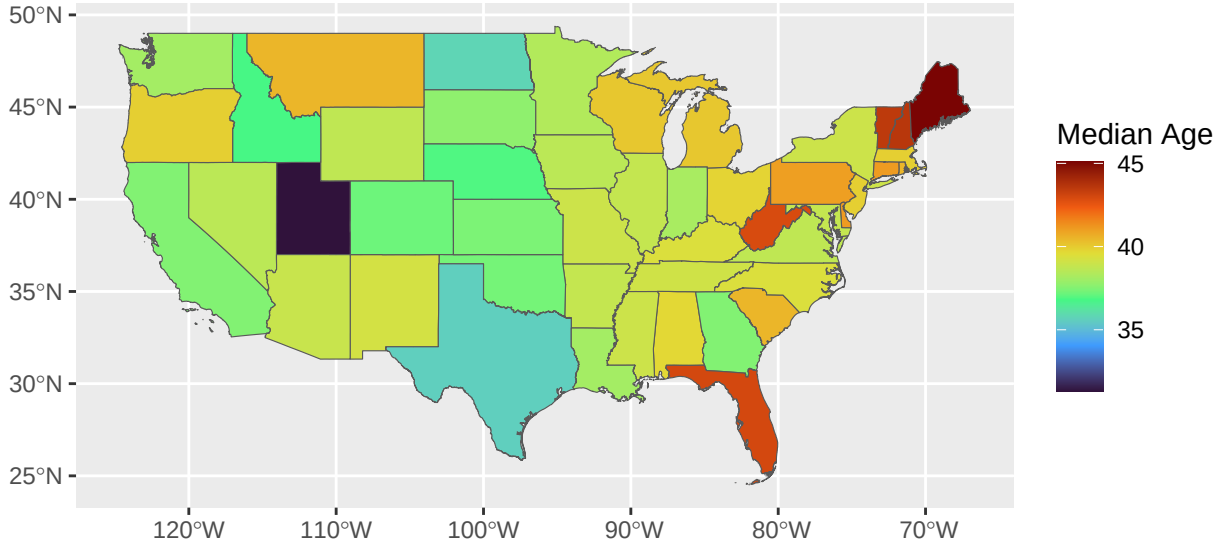
Spatial data in tidycensus

Our graph of the median age by state is interesting, but my mind immediately looks at that list of states and tries to determine if there's some kind of geographical trend. Fortunately, R is good at making maps using data.

```
age20 <- get_decennial(geography = "state",
                       variables = "P13_001N",
                       year = 2020,
                       sumfile = "dhc",
                       geometry = TRUE)
```

```
##   |                                                                              |
```

```
age20 %>%
  filter(!NAME %in% c("Alaska","Hawaii","Puerto Rico")) %>%
  ggplot(aes(fill = value))+
  geom_sf()+
  scale_fill_viridis_c(option = "turbo") +
  labs(fill= "Median Age")
```



Example relevant to our campaigns

So this is cool and all, but we are not going to be working on national presidential campaigns. Our campaigns are local, all in Hamilton County. Let's see if we can figure out how to make these same kinds of maps but on a local scale.

This time we will use American Community Survey (ACS) data because it is more detailed. We will stick with the recommendation in the book *Data Wrangling* (see above) to use the 5-year estimates as the"larger sample sizes gives them greater accuracy, particularly for smaller geographic units.''

We start by looking at the variable names for things that are available in the ACS data.

```
acs5.vars <- load_variables(2020, "acs5", cache = TRUE)
```

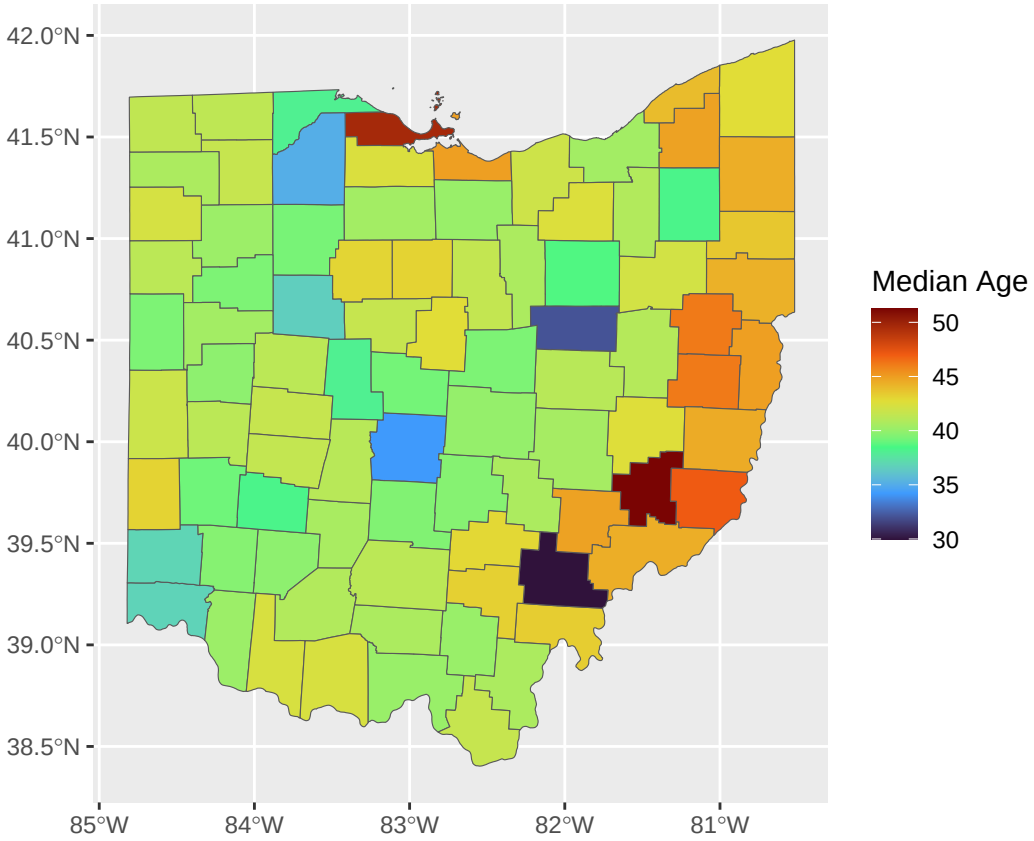
Search the data table, we find that the code "B01002_001" is used for median age in this data set.

Let's start by making the same map of median age, but this time just by Ohio's counties.

```
age20 <- get_acs(geography = "county",
                         state = "Ohio",
                         variables = "B01002_001",
                         year = 2020,
                         geometry = TRUE)
```

```
##   |                                                                              |
```

```
age20 %>%
  ggplot(aes(fill = estimate))+
  geom_sf()+
  scale_fill_viridis_c(option = "turbo") +
  labs(fill= "Median Age")
```



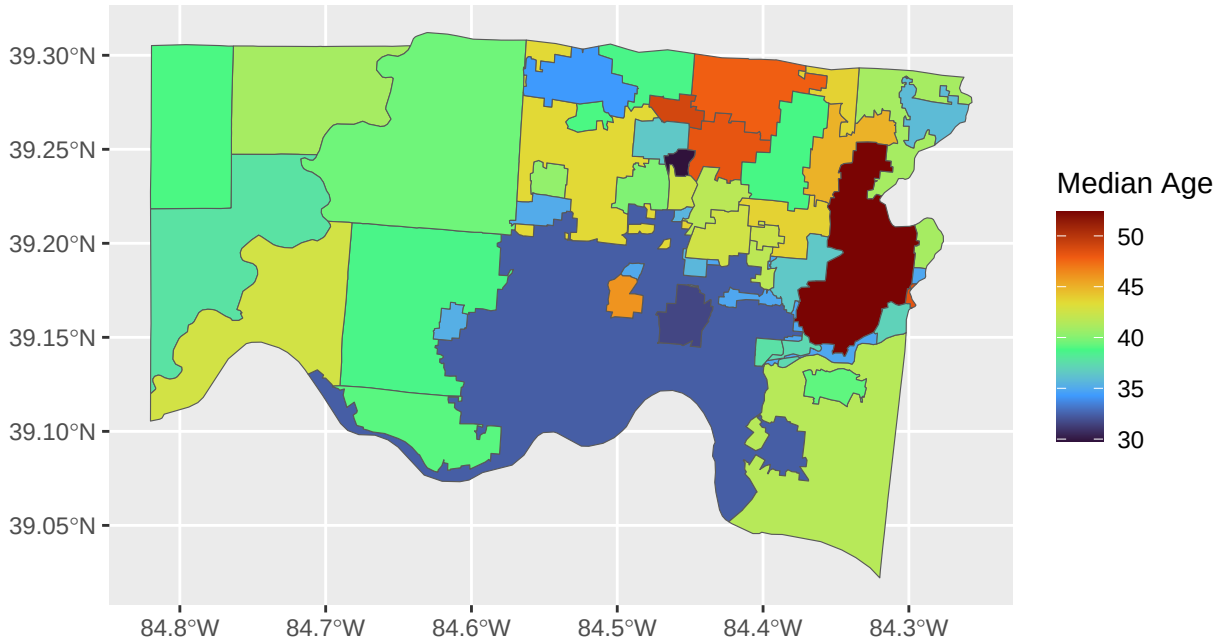
Let's go even down to a more local map. Let's look just inside Hamilton County.

```
age20 <- get_acs(geography = "county subdivision",
                 state = "Ohio",
                 county = "Hamilton",
```

```
                  variables = "B01002_001",
                  year = 2020,
                  geometry = TRUE)
```

```
##   |                                                                      |
```

```
age20 %>%
  ggplot(aes(fill = estimate))+
  geom_sf()+
  scale_fill_viridis_c(option = "turbo") +
  labs(fill= "Median Age")
```

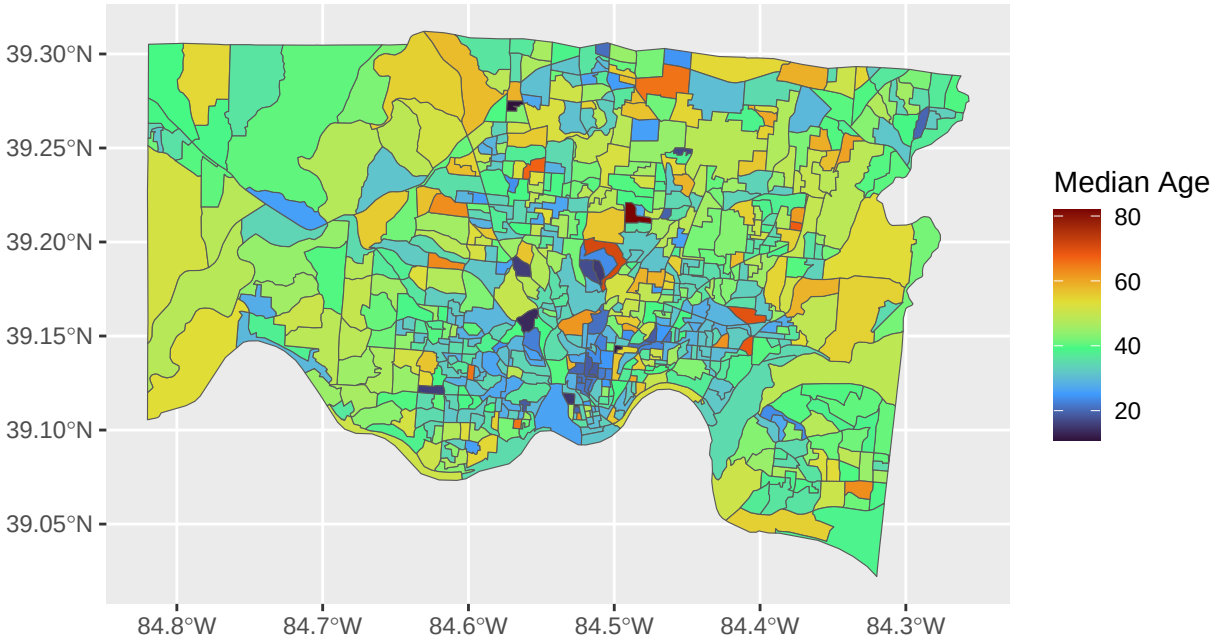


You have probably guessed the next step by now... let's go more local

```
age20 <- get_acs(geography = "block group",
                  state = "Ohio",
                  county = "Hamilton",
                  variables = "B01002_001",
                  year = 2020,
                  geometry = TRUE)
```

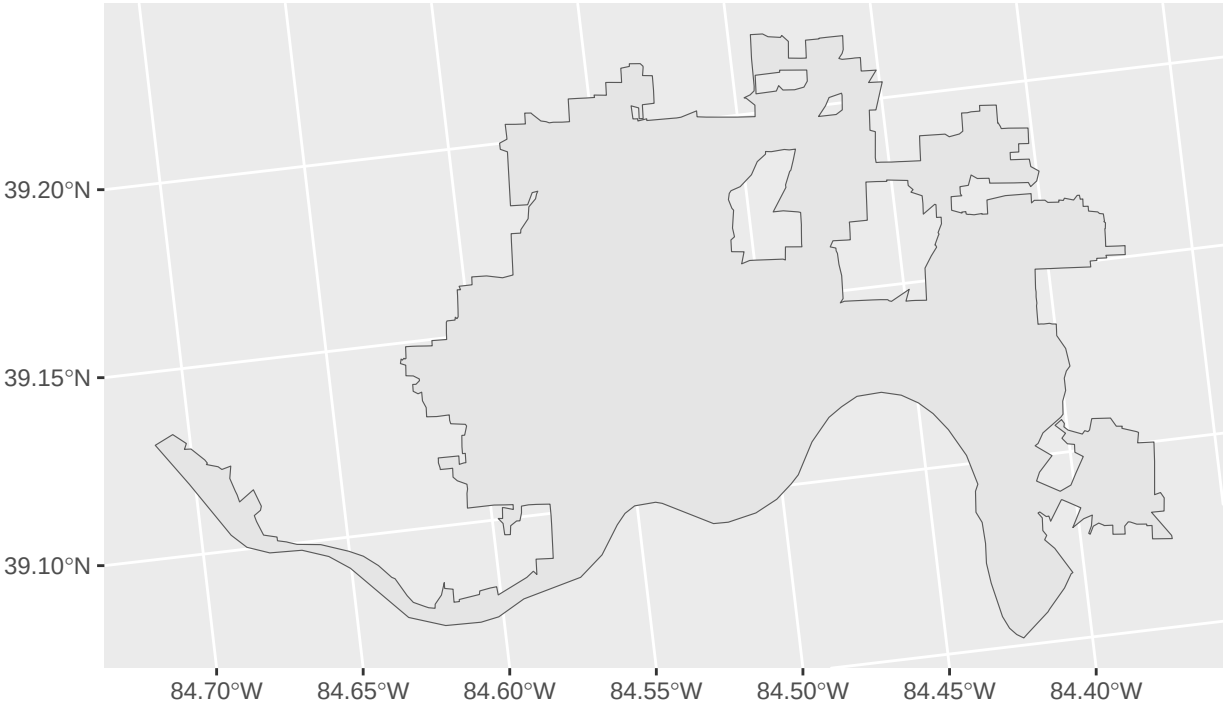
```
##   |                                                                      |
```

```
age20 %>%
  ggplot(aes(fill = estimate))+
  geom_sf()+
  scale_fill_viridis_c(option = "turbo") +
  labs(fill= "Median Age")
```

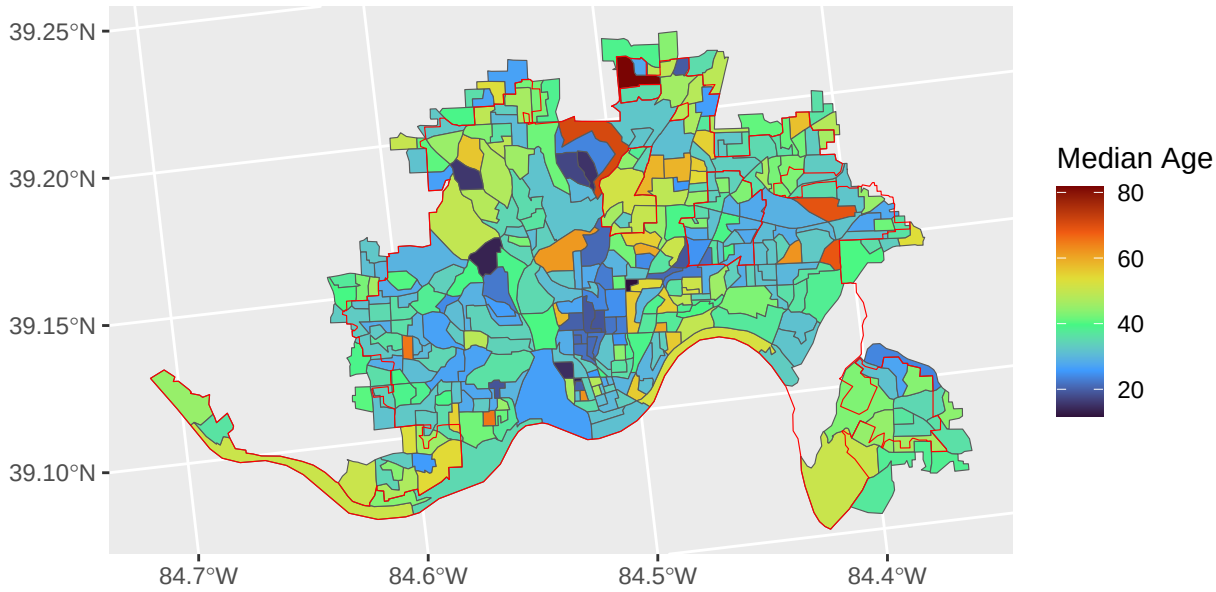


Our elections this year are for Cincinnati City Council. Let's just make a map of Cincinnati.

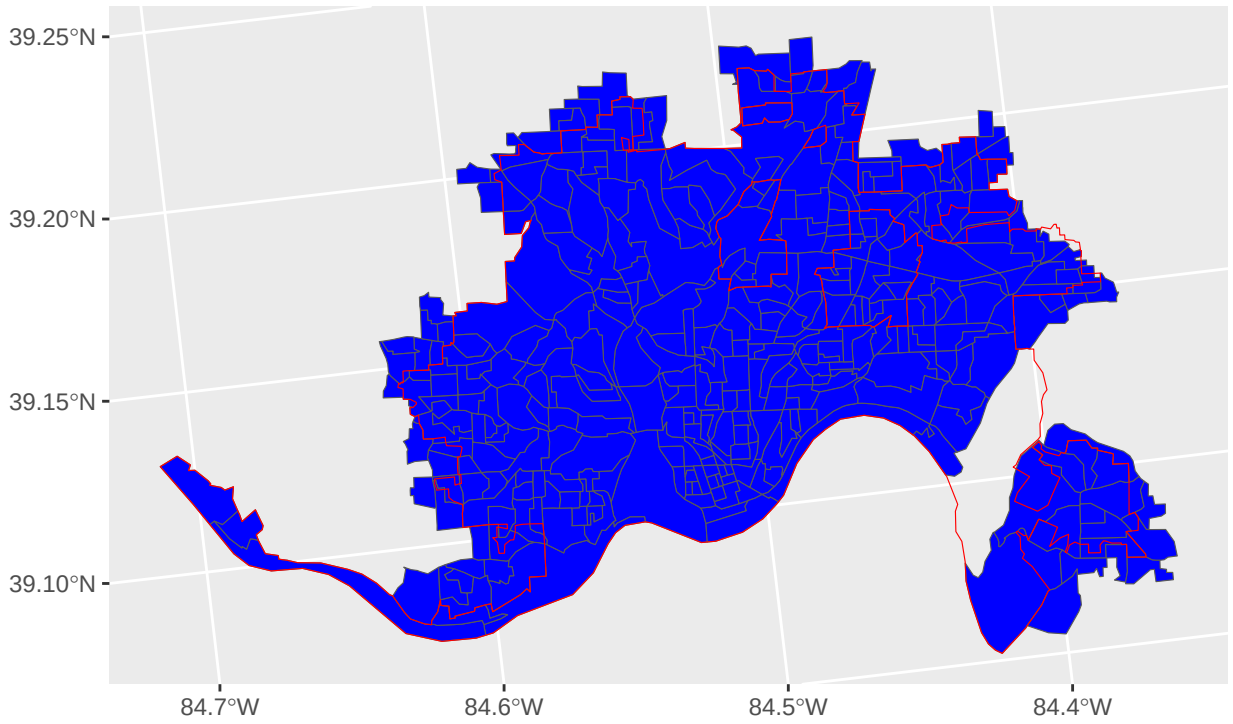
```
cincy <- get_acs(geography = "county subdivision",
                 state = "Ohio",
                 county = "Hamilton",
                 variables = "B01002_001",
                 year = 2020,
                 geometry = TRUE) %>%
  filter(NAME == "Cincinnati city, Hamilton County, Ohio") %>%
  st_transform(8528)

cincy %>%
  ggplot() +
  geom_sf()
```



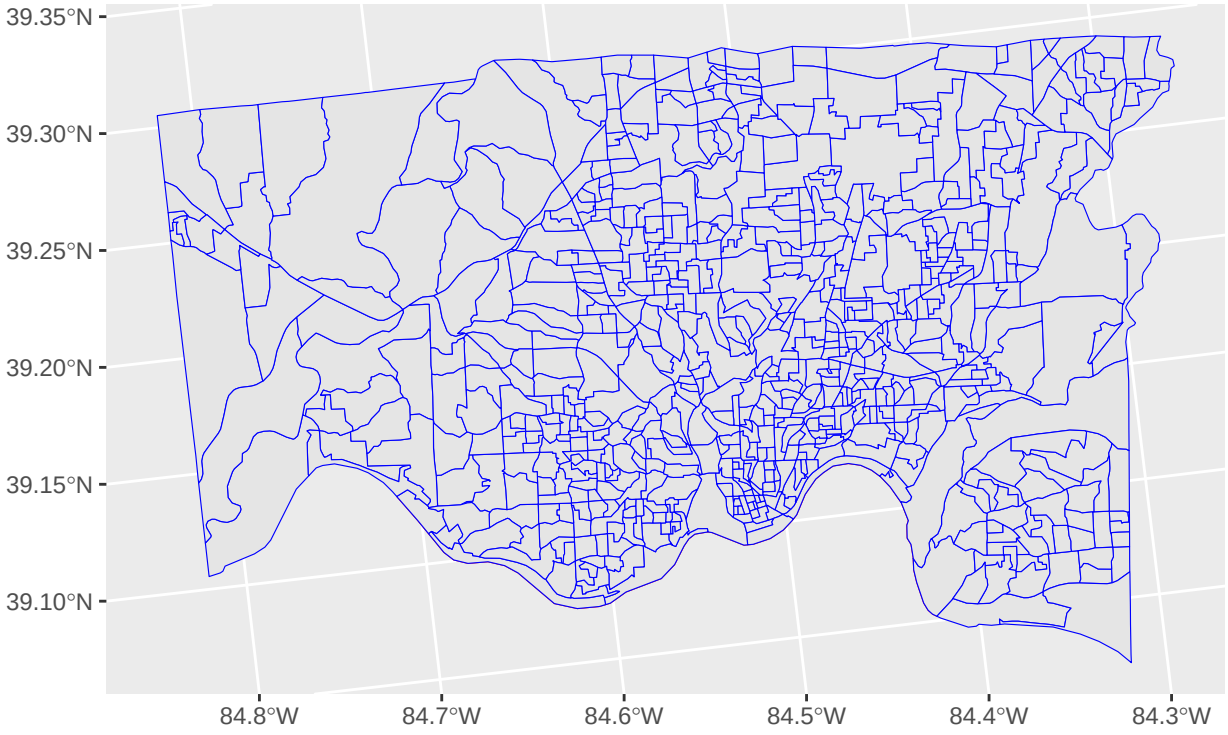

```
age20 <- get_acs(geography = "block group",
                 state = "Ohio",
                 county = "Hamilton",
                 variables = "B01002_001",
                 year = 2020,
                 geometry = TRUE) %>%
  st_transform(8528)

age20_cincy <- age20 %>%
  st_filter(st_buffer(cincy,4000),.predicate = st_within)

ggplot()+
  geom_sf(data = age20_cincy,aes(fill = estimate)) +
  geom_sf(data = cincy,color='red',fill=NA)+
  scale_fill_viridis_c(option = "turbo") +
  labs(fill= "Median Age")
```




```
ggplot() +
  geom_sf(data=age20_cincy,fill='blue') +
  geom_sf(data = cincy,color='red',fill=NA)
```




```
ggplot() +
  geom_sf(data = cincy,color='red',fill=NA) +
  geom_sf(data=age20,color="blue")
```


39.35°N
39.30°N
39.25°N
39.20°N
39.15°N
39.10°N
84.8°W
84.7°W
84.6°W
84.5°W
84.4°W
84.3°W