Accessing Median Income Data

August 1, 2022

Contents

Introduction

CBEP, like other National Estuary Programs will receive additional funding to support our programs via the "Bipartisan Infrastructure Law" signed into law last December.

EPA has recently released guidance for applying for those funds. A core component of the guidance is that overall, the NEP program should comply with the White House's "Justice 40" initiative, which requires that "at least 40% of the benefits and investments from BIL funding flow to disadvantaged communities."

EPA suggested that we use the National-scale EJSCREEN tools to help identify "disadvantaged communities" in our region. The EPA guidance goes on to suggest we focus on five demographic indicators:

- Percent low-income;
- Percent linguistically isolated;
- Percent less than high school education;
- Percent unemployed; and
- Low life expectancy.

These metrics may not make that much sense in our region, so we are exploring other types of metrics, including metrics based on median household income.

Here I merely document methods for accessing median household income data.

Load Libraries

```
library(tidyverse)
#> -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
#> v ggplot2 3.3.6     v purrr   0.3.4
#> v tibble  3.1.7     v dplyr   1.0.9
#> v tidyr   1.2.0     v stringr 1.4.0
#> v readr   2.1.2     v forcats 0.5.1
#> -- Conflicts ------------------------------------------ tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
library(GGally)
#> Registered S3 method overwritten by 'GGally':
#>   method from
#>   +.gg   ggplot2
library(readr)
```

Set Graphics Theme

This sets `ggplot()`graphics for no background, no grid lines, etc. in a clean format suitable for (some) publications.

```
theme_set(theme_classic())
```

Load Data

Folder References

I use folder references to allow limited indirection, thus making code from GitHub repositories more likely to run "out of the box".

```
data_folder <- "Original_Data"
dir.create(file.path(getwd(), 'figures'), showWarnings = FALSE)
```

Load Data

Base Data and Calculated Indexes

```
the_file <- "National_Draft_Indexes.csv"
the_data <- read_csv(the_file,
                     n_max = 74001,
                     col_types = c(rep(col_character(),3),
                                   rep(col_double(), 23))) %>%
  mutate(PCA_Index_V1 = -PCA_Index_V1,  # PCA is only unique to mirror images.
         PCA_Index_V2 = -PCA_Index_V2)
```

American Community Survey

The American Community Survey (and everything census!) data has completely obscure column names. The file and associated metadata includes enough info to get started. Data downloaded from here:

The file itself provides a little metadata in the second row. An accompanying metadata file provides a smidgen more information. Here I only download median household income, although the data set has a lot of other information on incomes.

```r
the_file <-'ACSST5Y2020.S1901_data_with_overlays_2022-05-06T151652.csv'
the_path <- file.path(data_folder, the_file)

inc_data <- read_csv(the_path, n_max = 85396,
                     col_types = cols_only('GEO_ID'= col_character(),
                                           'NAME'= col_character(),
                                           'S1901_C01_012E' = col_double(),
                                           'S1901_C01_012M' = col_double())) %>%
  filter(!row_number() == 1) %>%
  rename(Med_House_Inc = S1901_C01_012E,
         Med_House_Inc_err = S1901_C01_012M)
#> Warning: One or more parsing issues, see `problems()` for details
```

Parsing issues were caused by the second row, containing metadata.

I calculate the `GEOID10` to match other files, and extract the state from the NAME value.

```r
inc_data <- inc_data %>%
  mutate(NEG_MED_H_INC = (250000 - Med_House_Inc)/10000,
         GEOID10 = sub('1400000US','', GEO_ID),
         STATE = sub('Census Tract.*,.*, ','', NAME)
         ) %>%
  select(-STATE, -GEO_ID)
```

```r
the_data <- the_data %>%
  left_join(inc_data, by = 'GEOID10')
```

But notice that the income data is not available for quite a few census tracts. We retain only 82% of census tracts nationwide with data on percentage of low income people.

```r
round(sum(! is.na(the_data$Med_House_Inc))/ sum( ! is.na(the_data$LOWINCPCT)) * 100,3)
#> [1] 82.801
```

We do slightly better in Maine.

```r
tmp <- the_data %>%
  select(State, Med_House_Inc, LOWINCPCT) %>%
  filter(State == 'Maine')
round(sum(! is.na(tmp$Med_House_Inc))/ sum( ! is.na(tmp$LOWINCPCT)) * 100,3)
#> [1] 85.754
rm(tmp)
```

Still, it's not obvious that losing 15% of our cesus tracts is worthwhile.