Mapping Demographic Indexes for the Casco Bay Watershed

July 27, 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.

This notebook builds on the work in "Calc_CB_Indexes.pdf" to produce simple maps showing the distribution of these five demographic metrics and composite indexes based on national-scale percentiles on each.

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(readr)
#library(rgdal)
library(sf)    # automatically loads `sp` and `rgdal`
#> Linking to GEOS 3.9.1, GDAL 3.4.3, PROJ 7.2.1; sf_use_s2() is TRUE
#library(rgeos)  # Needed for polygon calcs without problematic dependency
#library(broom)  # used to tidy geospatial info to dataframe for ggplot
                 # but can use could use fortify() from ggplot as well.)
```

Set Graphics Theme

This sets `ggplot()`graphics for a blank slate, suitable for maps.

```
theme_set(theme_void())
```

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".

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

Tabular Data

```r
cb_data <- read_csv("cb_tracts_indexes.csv",
                    col_types = paste0('cdd--', rep('d', 23)))
```

Geospatial Data

We have geospatial data in UTM coordinates. Files are large, so not all are included in the github repository.

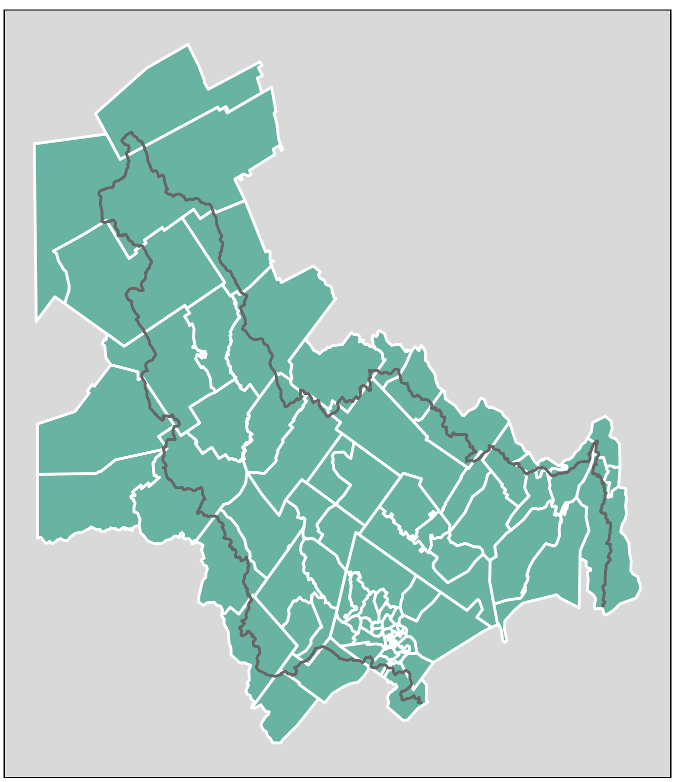
```r
the_file_name <- 'casco_ejscreen_utm.shp'
the_path <- file.path(gis_folder, the_file_name)
cb_geospatial <- st_read(the_path )
#> Reading layer `casco_ejscreen_utm' from data source
#>   `C:\Users\curtis.bohlen\Documents\Data Analysis\Casco_EJScreen\GIS_Data\casco_ejscreen_utm.shp'
#>   using driver `ESRI Shapefile'
#> Simple feature collection with 80 features and 8 fields
#> Geometry type: POLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 339455.5 ymin: 4817293 xmax: 438494.2 ymax: 4931318
#> Projected CRS: NAD83 / UTM zone 19N

 the_file_name <- 'casco_watershed_utm.shp'
 the_path <- file.path(gis_folder, the_file_name)
 cb_watershed <- st_read(the_path )
#> Reading layer `casco_watershed_utm' from data source
#>   `C:\Users\curtis.bohlen\Documents\Data Analysis\Casco_EJScreen\GIS_Data\casco_watershed_utm.shp'
#>   using driver `ESRI Shapefile'
#> Simple feature collection with 817 features and 9 fields
#> Geometry type: POLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 349981.6 ymin: 4823492 xmax: 433681.5 ymax: 4917055
#> Projected CRS: NAD83 / UTM zone 19N

the_file_name <- 'watershed_outline.shp'
the_path <- file.path(gis_folder, the_file_name)
cb_watershed_outline <- st_read(the_path )
#> Reading layer `watershed_outline' from data source
#>   `C:\Users\curtis.bohlen\Documents\Data Analysis\Casco_EJScreen\GIS_Data\watershed_outline.shp'
#>   using driver `ESRI Shapefile'
#> Simple feature collection with 1 feature and 1 field
#> Geometry type: LINESTRING
#> Dimension:     XY
#> Bounding box:  xmin: 349996 ymin: 4823674 xmax: 433678.3 ymax: 4917018
#> Projected CRS: NAD83 / UTM zone 19N

the_file_name <- 'Maine.shp'
```

```r
the_path <- file.path(gis_folder, the_file_name)
Maine <- st_read(the_path )
#> Reading layer `Maine' from data source
#>   `C:\Users\curtis.bohlen\Documents\Data Analysis\Casco_EJScreen\GIS_Data\Maine.shp'
#>   using driver `ESRI Shapefile'
#> Simple feature collection with 1 feature and 5 fields
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 336630.3 ymin: 4759246 xmax: 662458.3 ymax: 5256292
#> Projected CRS: NAD83 / UTM zone 19N
```

Initial Plot

Note that if you pass the geospatial layer to `ggplot` without manually converting it (using `tidy()` from `broom`, or `fortify()`, buried in `ggplot2`, `ggplot()` still figures out what to do, but makes some default decisions about how to define the polygons that may or may not be correct. For a sample map, as here, they appear to work pretty well.

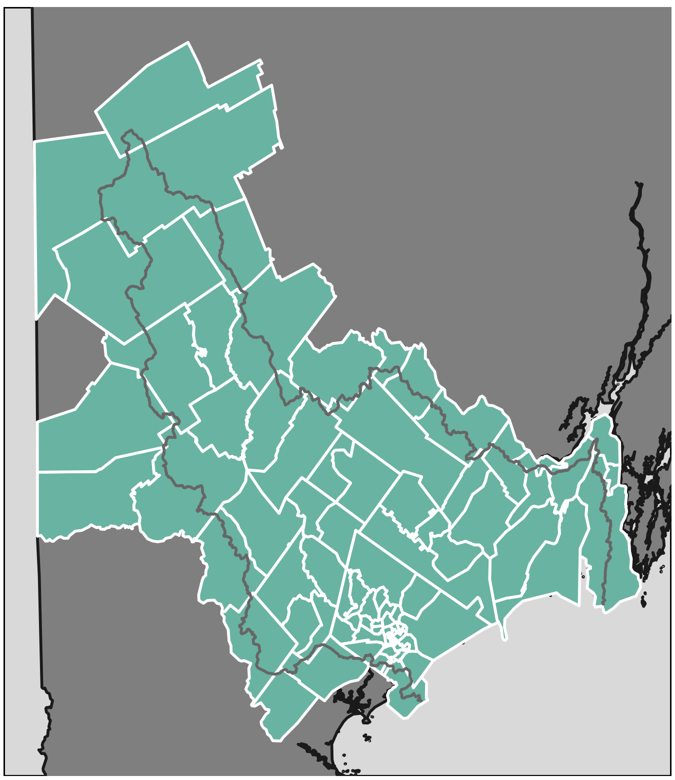
```r
plt <- ggplot() +
  geom_sf(data = cb_geospatial,
               fill="#69b3a2", color="white") +
  geom_sf(data = cb_watershed_outline,
            color="grey40") +

  theme(panel.background = element_rect(fill = 'grey85')) +
  coord_sf()
plt
```



That's a good start, but it would be nice to provide a Maine state background. To do that, we need to find the current coordinates and apply them after we add Maine (which is larger) to the map.

```
xlims <- layer_scales(plt)$x$range$range
ylims <- layer_scales(plt)$y$range$range
```

```
plt <- ggplot() +
  geom_sf(data = Maine,
           fill = 'grey50', color='grey10')  +
  geom_sf(data = cb_geospatial,
          fill="#69b3a2", color="white") +
  geom_sf(data = cb_watershed_outline,
          fill = NA, color="grey40") +


  theme(panel.background = element_rect(fill = 'grey85')) +
  coord_sf(xlim = xlims, ylim = ylims)
plt
```



Merge Tabular Data and Geospatial Data

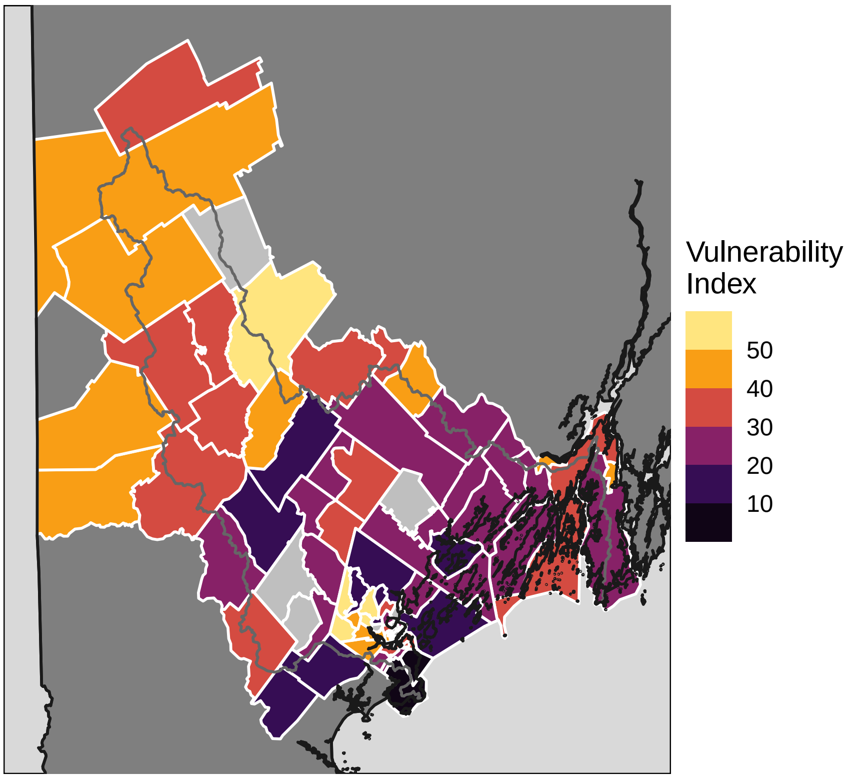
```
tmp <- cb_data %>%
  select(c(GEOID10, LIFEEXP:UNEMPPCT, -LIFEEXP_SE, NEG_LIFEEXP, Index_1:P_Index_2))

cb_geospatial_2<- cb_geospatial %>%
  left_join(tmp, by = 'GEOID10')
```

```
names(cb_geospatial_2)
#>  [1] "GEOID10"     "County"      "Shape_Leng"  "Shape_Area"  "InPoly_FID"
#>  [6] "SimPgnFlag"  "MaxSimpTol"  "MinSimpTol"  "LIFEEXP"     "NEG_LIFEEXP"
#> [11] "LOWINCPCT"   "LESSHSPCT"   "LINGISOPCT"  "UNEMPPCT"    "Index_1"
#> [16] "Index_2"     "P_Index_1"   "P_Index_2"   "geometry"
```

Revised Plot

With several other improvements...

```
plt <- ggplot() +
  geom_sf(data = Maine,
              fill = 'grey50',
             color='grey10')  +
  geom_sf(data = cb_geospatial_2, aes(fill = Index_2), color="white") +
  geom_sf(data = cb_watershed_outline,
          color="grey40") +
  geom_sf(data = Maine,  fill = NA,
              color='grey10')  +
  scale_fill_viridis_b(option = 'B', na.value = "grey75",
                       name = 'Vulnerability\nIndex') +

  theme(panel.background = element_rect(fill = 'grey85')) +
  coord_sf(xlim = xlims, ylim = ylims)
plt
```



Creating A Faceted Map Display

Building the Tibble

To create a `facet_wrap()` display on multiple panels, I need to build a suitable long tibble.

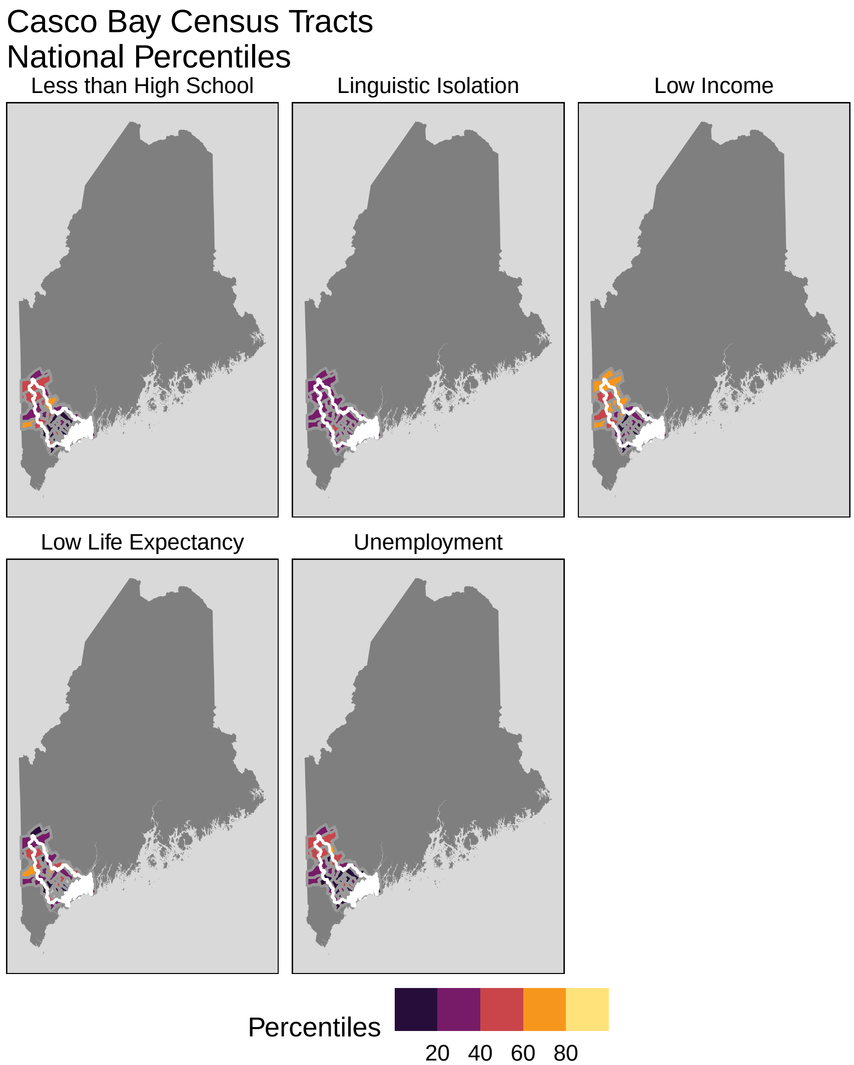
```
tmp <- cb_data %>%
  select(GEOID10, P_NEG_LIFEEXP:P_UNEMPPCT) %>%
rename(`Low Life Expectancy` = P_NEG_LIFEEXP,
       `Low Income` = P_LWINCPCT,
       `Less than High School` = P_LESHSPCT,
       `Linguistic Isolation` = P_LNGISPCT,
       `Unemployment` = P_UNEMPPCT)
```

```
cb_geospatial_3 <- cb_geospatial %>%
  left_join(tmp, by = 'GEOID10') %>%
  pivot_longer(`Low Life Expectancy`:`Unemployment`,
               names_to = 'Threshold', values_to = 'Value')
```

Whole Watershed Facet Plot

This ggplot object is fairly complex, so it takes some time to render. Rendering time can be shortened somewhat by using simpler polygon geometries.

```
plt <- ggplot() +
  geom_sf(data = Maine,
              fill = 'grey50', color=NA)  +
  geom_sf(data = cb_geospatial_3, aes(fill = Value), color="grey60") +


  facet_wrap(~ Threshold, nrow = 2) +

  scale_fill_viridis_b(option = 'B', name = 'Percentiles',
                       na.value = "grey75", breaks = c(0, 20, 40, 60, 80,  100) ) +

  theme(panel.background = element_rect(fill = 'grey85'),
        legend.position = 'bottom',
        strip.text.x = element_text(margin = margin(0,0,0.1,0, "cm"))) +
  coord_sf(xlim = xlims, ylim = ylims) +
  ggtitle("Casco Bay Census Tracts\nNational Percentiles")

plt +
  geom_sf(data = cb_watershed, fill = NA,
             color="white")
#> Coordinate system already present. Adding new coordinate system, which will replace the existing one
```

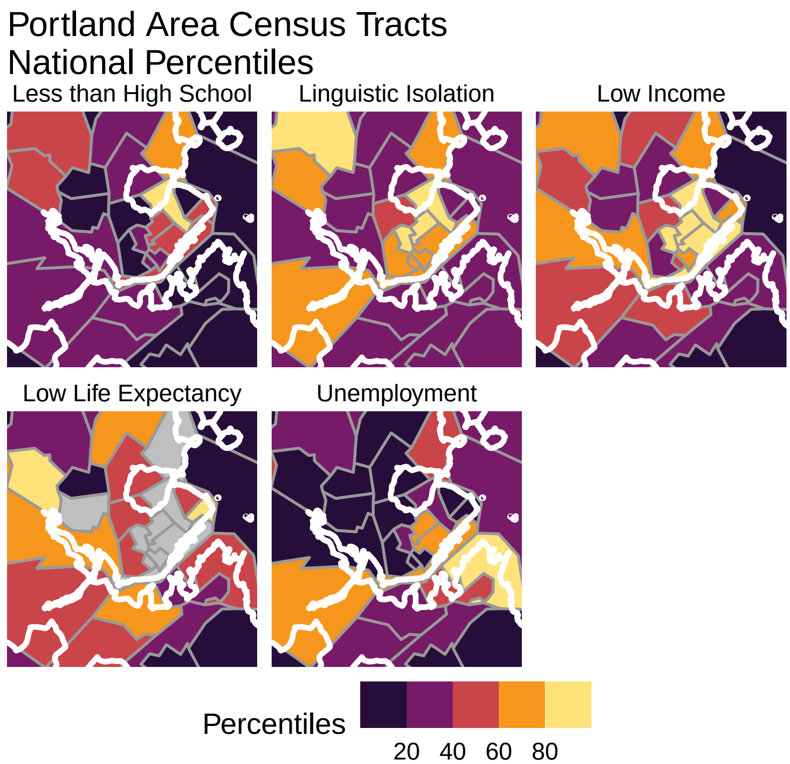



```
ggsave('figures/watershed_national.png', type='cairo',
         width = 7, height = 6)
ggsave('figures/watershed_national.pdf', device = cairo_pdf,
       width = 7, height = 6)
```

Zoom in on Portland Region

```
new_x_low  <- xlims[1] + 13 * (xlims[2]-xlims[1])/24
new_x_high <- xlims[1] + 15 * (xlims[2]-xlims[1])/24

new_y_low  <- ylims[1] + 5 * (ylims[2]-ylims[1])/44
new_y_high <- ylims[1] + 9 *(ylims[2]-ylims[1])/48

new_xlims  <- c(new_x_low, new_x_high)
new_ylims  <- c(new_y_low, new_y_high)
```

```r
plt +
  geom_sf(data = cb_watershed, fill = NA,
              color="white", lwd = 1) +
  coord_sf(xlim = new_xlims, ylim = new_ylims) +
  ggtitle("Portland Area Census Tracts\nNational Percentiles")
#> Coordinate system already present. Adding new coordinate system, which will replace the existing one
```




```r
ggsave('figures/portland_national.png', type='cairo',
         width = 7, height = 6)
ggsave('figures/portland_national.pdf', device = cairo_pdf,
       width = 7, height = 6)
```

So, several metrics pick up the subsidized housing and homeless community in Portland's Back Cove neighborhood. Other than that, what is striking is the fact that the ranks of the different metrics are not all that correlated in this smaller sub-region, which may help explain why few locations are flagged by any of the candidate indicators.

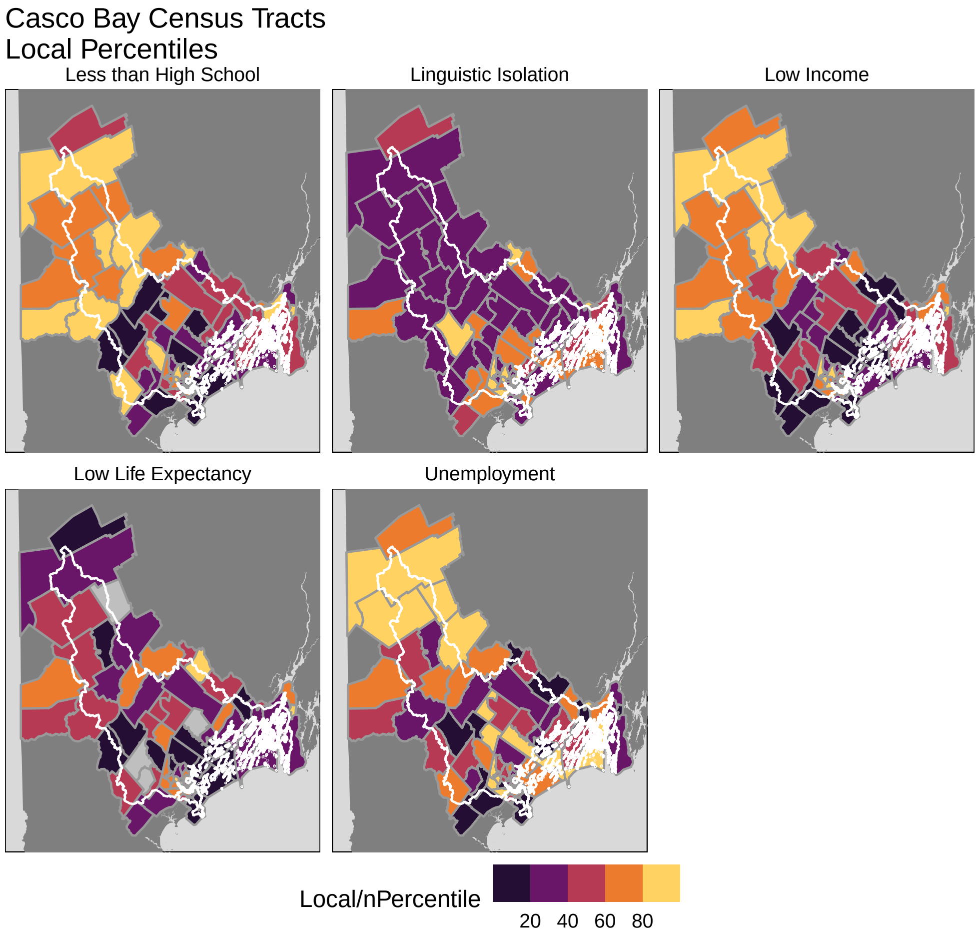
Observed Percentiles of Indicators

Let's look at percentiles within our region. I calculate these by calculating ranks, and dividing my the overall sample size.

Whole Watershed

```
tmp <- cb_data %>%
  select(GEOID10, LOWINCPCT:UNEMPPCT, NEG_LIFEEXP) %>%
  mutate(across(LOWINCPCT:NEG_LIFEEXP, function(x)
                                       rank(x, na.last = 'keep',
                                            ties.method = 'average')/81)*100) %>%
rename(`Low Life Expectancy` = NEG_LIFEEXP,
       `Low Income` = LOWINCPCT,
       `Less than High School` = LESSHSPCT,
       `Linguistic Isolation` = LINGISOPCT,
       `Unemployment` = UNEMPPCT)

cb_geospatial_4 <- cb_geospatial %>%
  left_join(tmp, by = 'GEOID10')

cb_geospatial_5 <- cb_geospatial_4 %>%
  pivot_longer(`Low Income`:`Low Life Expectancy`,
               names_to = 'Indicator', values_to = 'Value')
```

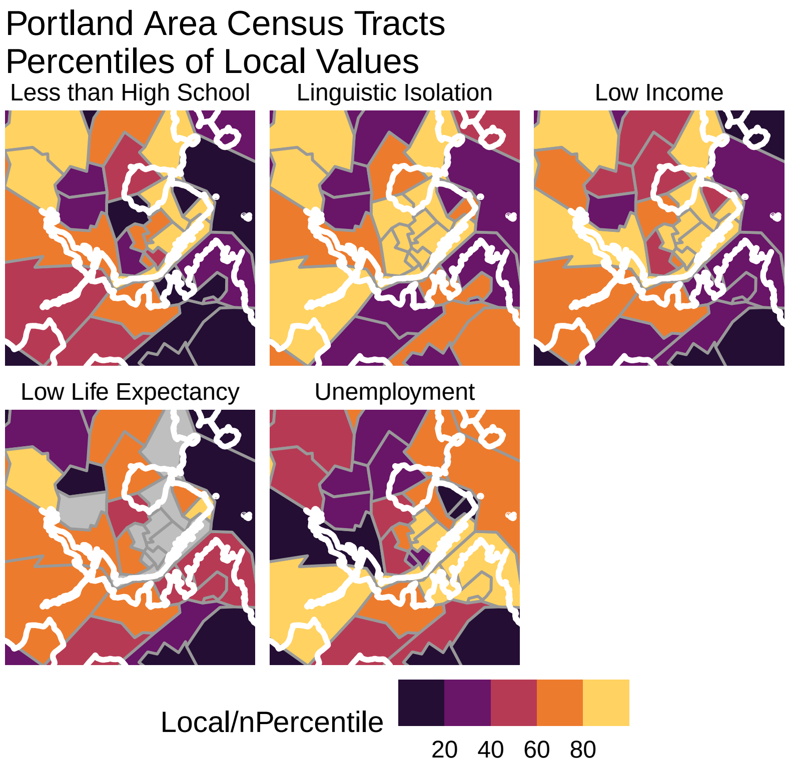
```
plt <- ggplot() +
  geom_sf(data = Maine,
                fill = 'grey50', color=NA)  +
  geom_sf(data = cb_geospatial_5, aes(fill = Value), color='grey60') +
  geom_sf(data = cb_watershed,  color="white", fill = NA) +

  facet_wrap(~ Indicator, nrow = 2) +

  scale_fill_viridis_b(option = 'B',  name = 'Local/nPercentile',
                       na.value = "grey75", breaks = c(0, 20, 40, 60, 80, 100 )) +

  theme(panel.background = element_rect(fill = 'grey85'),
        legend.position = 'bottom',
        strip.text.x = element_text(margin = margin(0,0,0.1,0, "cm"))) +
  coord_sf(xlim = xlims, ylim = ylims) +
  ggtitle("Casco Bay Census Tracts\nLocal Percentiles")
plt
```




```
ggsave('figures/watershed_regional.png', type='cairo',
         width = 7, height = 6)
ggsave('figures/watershed_regional.pdf', device = cairo_pdf,
       width = 7, height = 6)
```

Portland Region

```
 plt +
  geom_sf(data = cb_watershed, fill = NA,
              color="white", lwd = 1) +
  coord_sf(xlim = new_xlims, ylim = new_ylims) +
   ggtitle("Portland Area Census Tracts\nPercentiles of Local Values")
#> Coordinate system already present. Adding new coordinate system, which will replace the existing one
```




```
ggsave('figures/portland_regional.png', type='cairo',
         width = 7, height = 6)
ggsave('figures/portland_regional.pdf', device = cairo_pdf,
       width = 7, height = 6)
```