Wrap-up
Loose ends

Maithreyi Gopalan

Week 9

```
#install.packages("Cairo")
```

Agenda

- Wrap up maps and loose ends with HTML

Let's get some census data

Note

To do this, you need to first register an API key with the US Census, which you can do here. Then use `census_api_key("YOUR API KEY")`.

Alternatively, you can specify `CENSUS_API_KEY = "YOUR API KEY"` in **.Renviron**. You can do this by using `usethis::edit_r_environ()`

Getting the data

```
library(tidycensus)
# Find variable code
# v <- load_variables(2019, "acs5")
# View(v)

census_vals <- get_acs(
  geography = "tract",
  state = "OR",
  variables = c(med_income = "B06011_001",
                ed_attain = "B15003_001"),
  year = 2019,
  geometry = TRUE
)
```

```
##
  |
  |
  |
  |==
  |
  |=====
  |
  |=======
  |
```

Look at the data

```
census_vals
```

```
## Simple feature collection with 1668 features and 5 fields (with 12 geometries e
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -124.5662 ymin: 41.99179 xmax: -116.4635 ymax: 46.29204
## Geodetic CRS:  NAD83
## First 10 features:
##          GEOID                                          NAME   variable estima
## 1  41031960302 Census Tract 9603.02, Jefferson County, Oregon med_income    300
## 2  41031960302 Census Tract 9603.02, Jefferson County, Oregon  ed_attain     30
## 3  41057960100    Census Tract 9601, Tillamook County, Oregon med_income    302
## 4  41057960100    Census Tract 9601, Tillamook County, Oregon  ed_attain     27
## 5  41015950100        Census Tract 9501, Curry County, Oregon med_income    189
## 6  41015950100        Census Tract 9501, Curry County, Oregon  ed_attain     23
## 7  41039001902        Census Tract 19.02, Lane County, Oregon med_income    248
## 8  41039001902        Census Tract 19.02, Lane County, Oregon  ed_attain     40
## 9  41029000300         Census Tract 3, Jackson County, Oregon med_income    250
## 10 41029000300         Census Tract 3, Jackson County, Oregon  ed_attain     43
```

Remove missing geometry rows

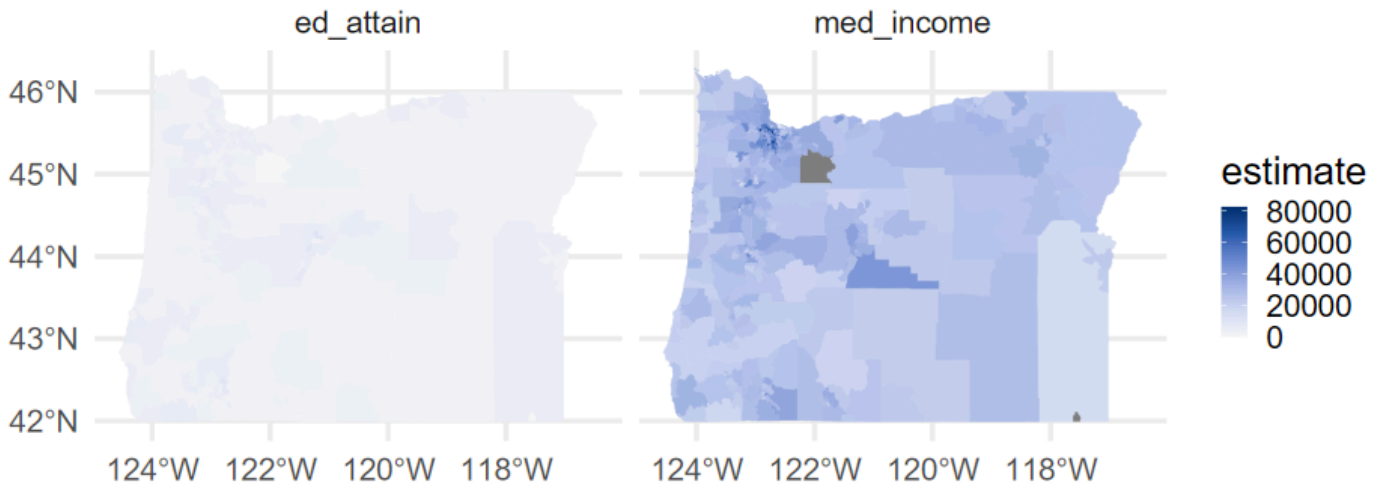
- Tidycensus is (currently) bringing in some rows with missing geometries
- This is not a big deal for ggplot, but is for other plotting systems
- Let's remove those rows

```
census_vals <- census_vals[!st_is_empty(census_vals$geometry), ,
```

Plot it

```
library(colorspace)
ggplot(census_vals) +
  geom_sf(aes(fill = estimate, color = estimate)) +
  facet_wrap(~variable) +
  guides(color = "none") +
  scale_fill_continuous_diverging("Blue-Red 3", rev = TRUE) +
  scale_color_continuous_diverging("Blue-Red 3", rev = TRUE)
```

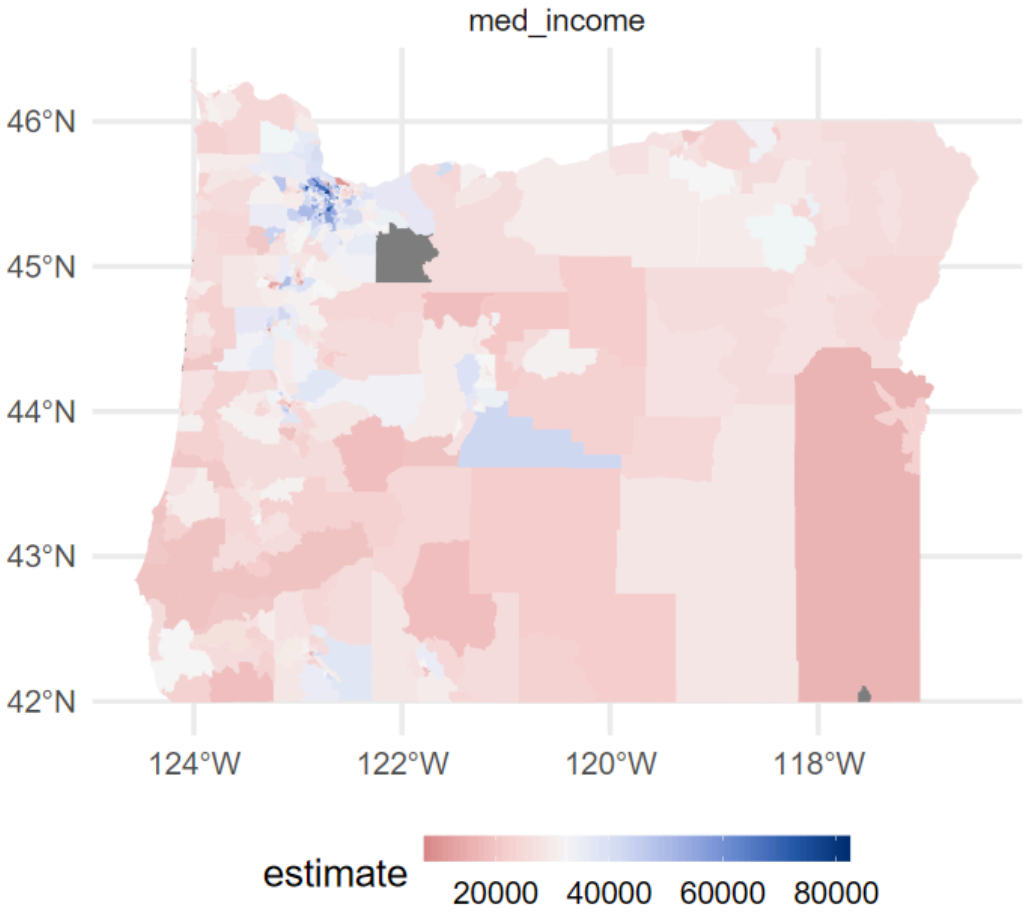
hmm...


ed_attain
med_income
46°N
45°N
44°N
43°N
42°N
124°W
122°W
120°W
118°W
estimate
80000
60000
40000
20000
0

Try again

```
library(colorspace)
income <- filter(census_vals, variable == "med_income")

income_plot <- ggplot(income) +
  geom_sf(aes(fill = estimate, color = estimate)) +
  facet_wrap(~variable) +
  guides(color = "none") +
  scale_fill_continuous_diverging(
    "Blue-Red 3",
    rev = TRUE,
    mid = mean(income$estimate, na.rm = TRUE)
  ) +
  scale_color_continuous_diverging(
    "Blue-Red 3",
    rev = TRUE,
    mid = mean(income$estimate, na.rm = TRUE)
  ) +
  theme(legend.position = "bottom",
        legend.key.width = unit(2, "cm"))
```

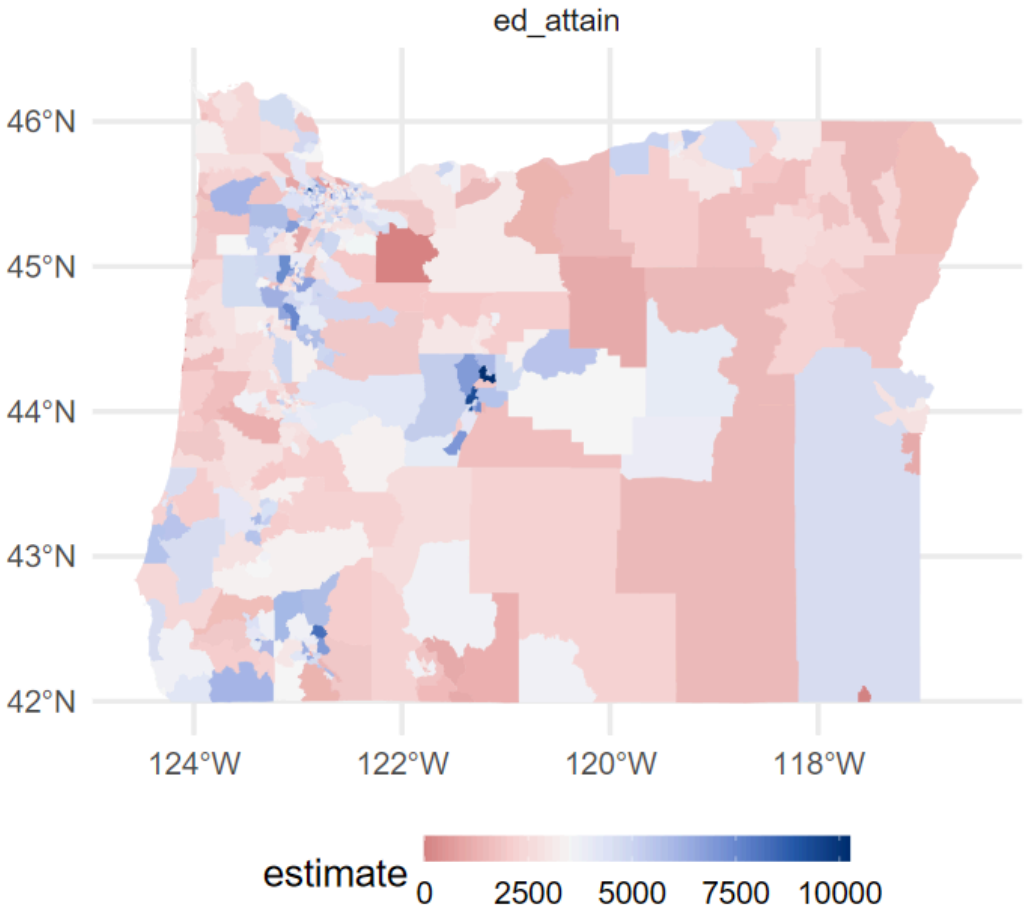
```
income_plot
```


med_income
46°N
45°N
44°N
43°N
42°N
124°W
122°W
120°W
118°W
estimate
20000
40000
60000
80000

Same thing for education

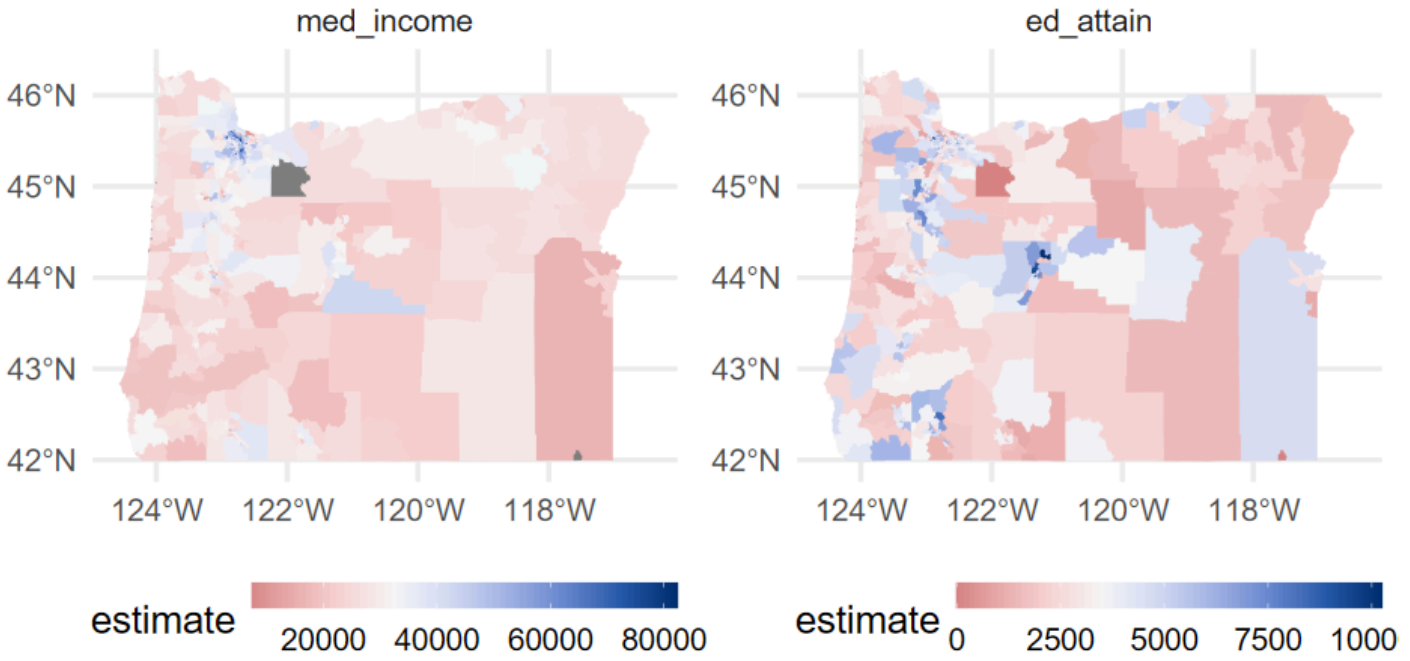
```
ed <- filter(census_vals, variable == "ed_attain")

ed_plot <- ggplot(ed) +
  geom_sf(aes(fill = estimate, color = estimate)) +
  facet_wrap(~variable) +
  guides(color = "none") +
  scale_fill_continuous_diverging(
    "Blue-Red 3",
    rev = TRUE,
    mid = mean(ed$estimate, na.rm = TRUE)
  ) +
  scale_color_continuous_diverging(
    "Blue-Red 3",
    rev = TRUE,
    mid = mean(ed$estimate, na.rm = TRUE)
  ) +
  theme(legend.position = "bottom",
        legend.key.width = unit(2, "cm"))
```

```
ed_plot
```


ed_attain
46°N
45°N
44°N
43°N
42°N
124°W
122°W
120°W
118°W
estimate
0
2500
5000
7500
10000

Put them together

```
gridExtra::grid.arrange(income_plot, ed_plot, ncol = 2)
```

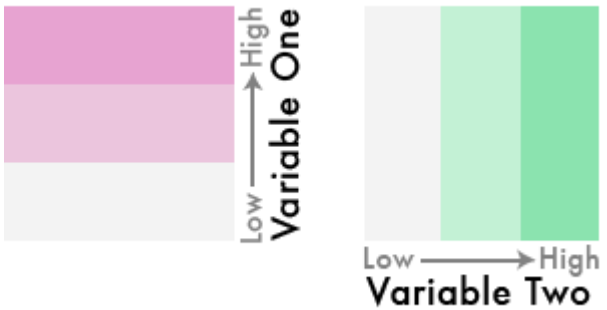


Bivariate color scales

How?

There are a few different ways. Here's one:

- Break continuous variable into categorical values
- Assign each combination of values between categorical vars a color
- Make sure the combinations of the colors make sense



gif from Joshua Stevens

Do it

Note - this will be fairly quick. I'm not expecting you to know how to do this, but I want to show you the idea and give you the breadcrumbs for the code you may need.

First - move it to wider

```
wider <- get_acs(
  geography = "tract",
  state = "OR",
  variables = c(med_income = "B06011_001",
                ed_attain = "B15003_001"),
  year = 2019,
  geometry = TRUE,
  output = "wide"
)

# remove missing geometry rows
wider <- wider[!st_is_empty(wider$geometry), , drop = FALSE]
```

Find the quartiles

```
ed_quartiles <- quantile(
  wider$ed_attainE,
  probs = seq(0, 1, length.out = 4),
  na.rm = TRUE
)

inc_quartiles <- quantile(
  wider$med_incomeE,
  probs = seq(0, 1, length.out = 4),
  na.rm = TRUE
)

ed_quartiles
```

```
##        0% 33.33333% 66.66667%      100%
##     0.000  2715.000  3949.333 10245.000
```

```
inc_quartiles
```

```
##        0% 33.33333% 66.66667%      100%
##   7449.00  26718.33  34375.00  82375.00
```

Create the cut variable

```
wider <- wider %>%
  mutate(cat_ed = cut(ed_attainE, ed_quartiles),
         cat_inc = cut(med_incomeE, inc_quartiles))
wider %>%
  select(starts_with("cat"))
```

```
## Simple feature collection with 828 features and 2 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -124.5662 ymin: 41.99179 xmax: -116.4635 ymax: 46.29204
## Geodetic CRS:  NAD83
## First 10 features:
##                  cat_ed             cat_inc                       geometry
## 1  (2.71e+03,3.95e+03] (2.67e+04,3.44e+04] MULTIPOLYGON (((-121.8495 4...
## 2         (0,2.71e+03] (2.67e+04,3.44e+04] MULTIPOLYGON (((-123.983 45...
## 3         (0,2.71e+03] (7.45e+03,2.67e+04] MULTIPOLYGON (((-124.5646 4...
## 4  (3.95e+03,1.02e+04] (7.45e+03,2.67e+04] MULTIPOLYGON (((-122.9855 4...
## 5  (3.95e+03,1.02e+04] (7.45e+03,2.67e+04] MULTIPOLYGON (((-122.9079 4...
## 6  (2.71e+03,3.95e+03] (3.44e+04,8.24e+04] MULTIPOLYGON (((-123.5016 4...
## 7         (0,2.71e+03] (7.45e+03,2.67e+04] MULTIPOLYGON (((-123.0927 4...
## 8  (2.71e+03,3.95e+03] (7.45e+03,2.67e+04] MULTIPOLYGON (((-120.8811 4...
## 9  (2.71e+03,3.95e+03] (7.45e+03,2.67e+04] MULTIPOLYGON (((-123.5093 4...
## 10 (2.71e+03,3.95e+03] (7.45e+03,2.67e+04] MULTIPOLYGON (((-123.8179 4...
```

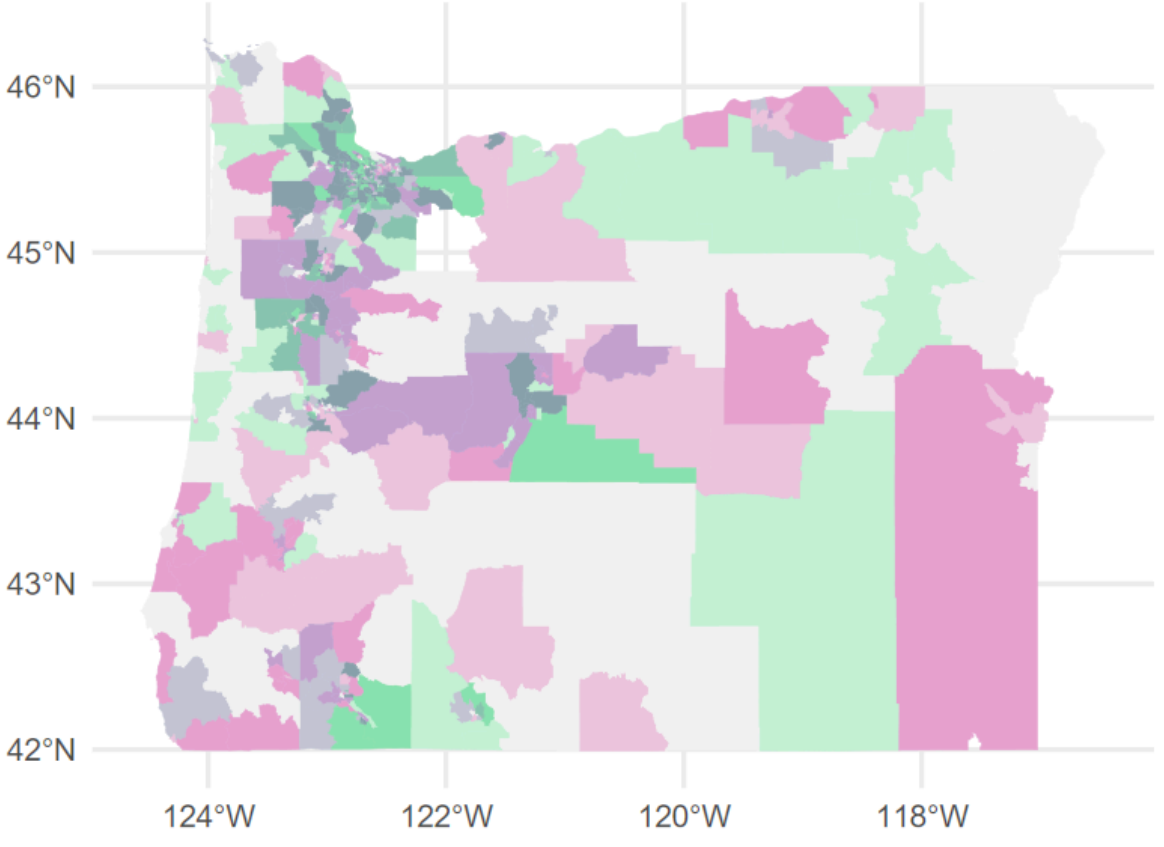
Set palette

```
# First drop geo column
pal <- st_drop_geometry(wider) %>%
  count(cat_ed, cat_inc) %>%
  arrange(cat_ed, cat_inc) %>%
  drop_na(cat_ed, cat_inc) %>%
  mutate(pal = c("#F3F3F3", "#C3F1D5", "#8BE3AF",
                 "#EBC5DD", "#C3C5D5", "#8BC5AF",
                 "#E7A3D1", "#C3A3D1", "#8BA3AE"))
pal
```

```
##                cat_ed             cat_inc   n     pal
## 1        (0,2.71e+03] (7.45e+03,2.67e+04] 116 #F3F3F3
## 2        (0,2.71e+03] (2.67e+04,3.44e+04]  85 #C3F1D5
## 3        (0,2.71e+03] (3.44e+04,8.24e+04]  70 #8BE3AF
## 4 (2.71e+03,3.95e+03] (7.45e+03,2.67e+04]  87 #EBC5DD
## 5 (2.71e+03,3.95e+03] (2.67e+04,3.44e+04]  97 #C3C5D5
## 6 (2.71e+03,3.95e+03] (3.44e+04,8.24e+04]  92 #8BC5AF
## 7 (3.95e+03,1.02e+04] (7.45e+03,2.67e+04]  71 #E7A3D1
## 8 (3.95e+03,1.02e+04] (2.67e+04,3.44e+04]  92 #C3A3D1
## 9 (3.95e+03,1.02e+04] (3.44e+04,8.24e+04] 113 #8BA3AE
```

Join & plot

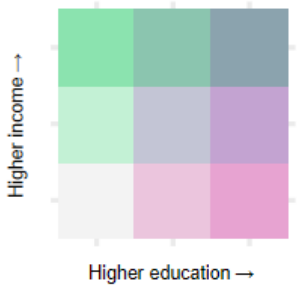
```
bivar_map <- left_join(wider, pal) %>%
  ggplot() +
  geom_sf(aes(fill = pal, color = pal)) +
  guides(fill = "none", color = "none") +
  scale_fill_identity() +
  scale_color_identity()
```


46°N
45°N
44°N
43°N
42°N
124°W
122°W
120°W
118°W

Add in legend

First create it

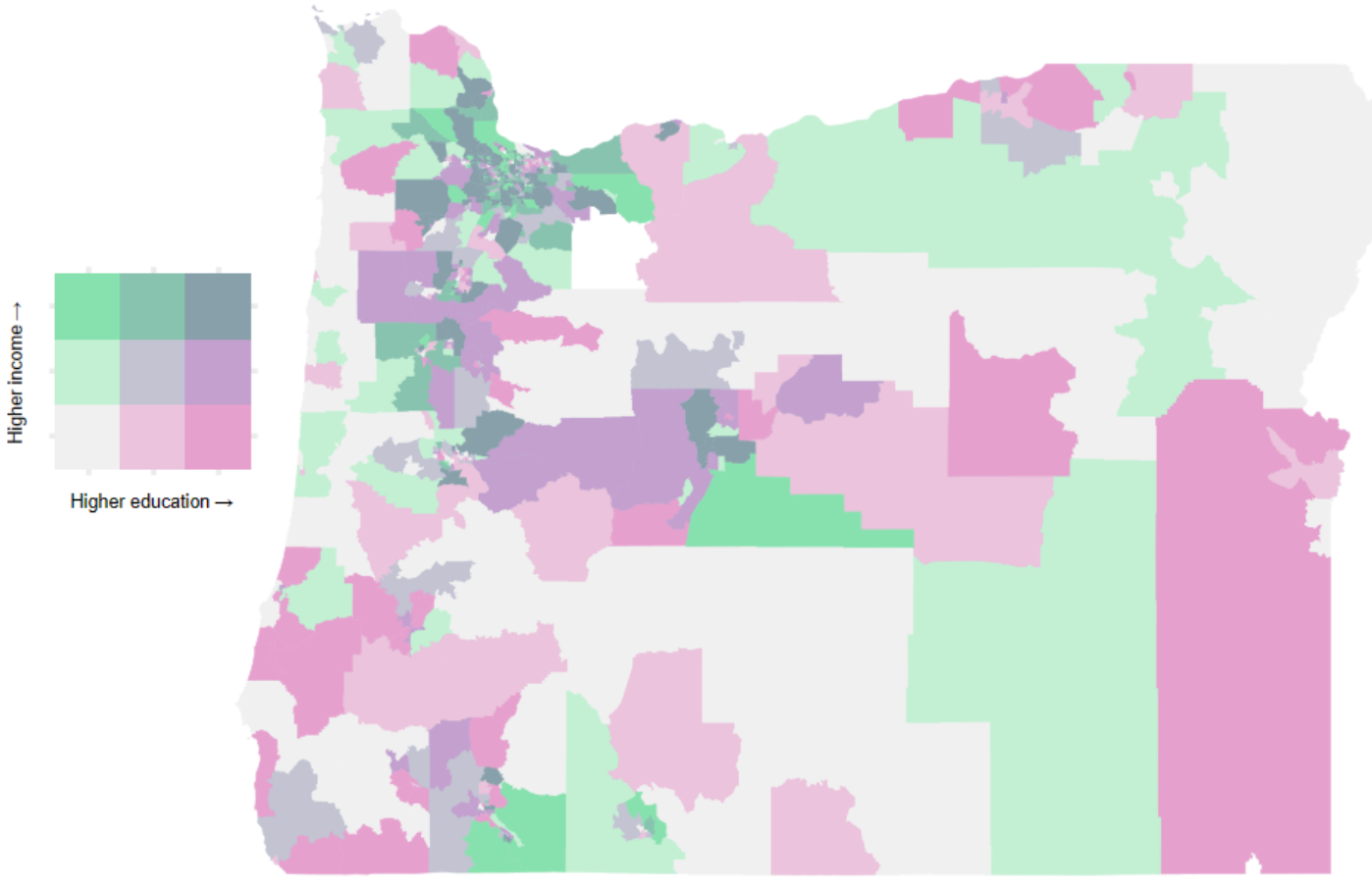
```
leg <- ggplot(pal, aes(cat_ed, cat_inc)) +
  geom_tile(aes(fill = pal)) +
  scale_fill_identity() +
  coord_fixed() +
  labs(x = expression("Higher education" %->% ""),
       y = expression("Higher income" %->% "")) +
  theme(axis.text = element_blank(),
        axis.title = element_text(size = 12))
leg
```



Put together

```
library(cowplot)
ggdraw() +
  draw_plot(bivar_map + theme_void(), 0.1, 0.1, 1, 1) +
  draw_plot(leg, -0.05, 0, 0.3, 0.3)
```

Coordinates are mostly guess/check depending on aspect ratio


Higher income →
Higher education →

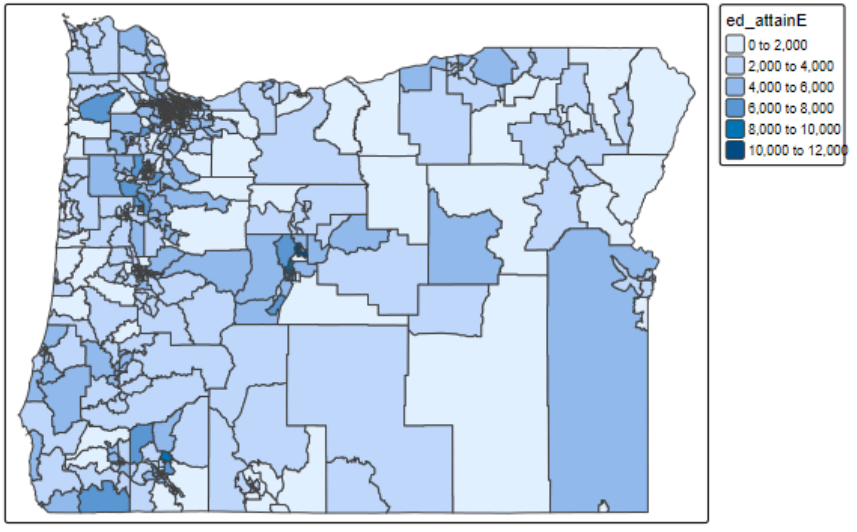
{tmap}

Back to just one variable

I mostly use `ggplot()`, but the **{tmap}** package is really powerful and the syntax is pretty straightforward, so let's have a quick overview.

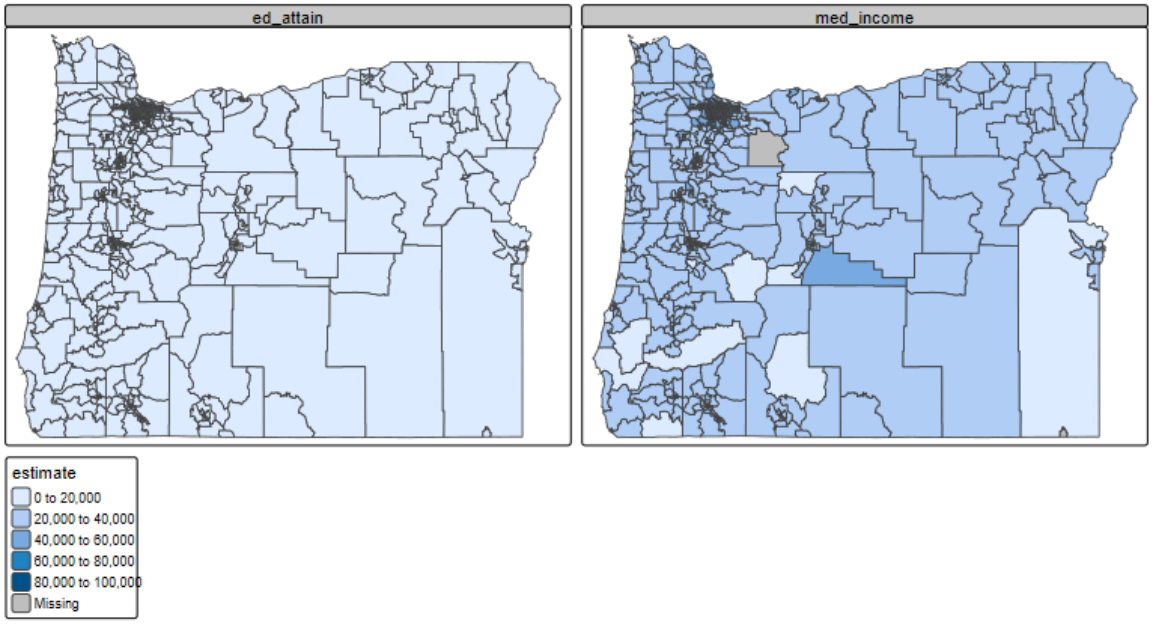
Education map with {tmap}.

```
library(tmap)
tm_shape(wider) +
  tm_polygons("ed_attainE") +
  tm_layout(legend.outside = TRUE)
```



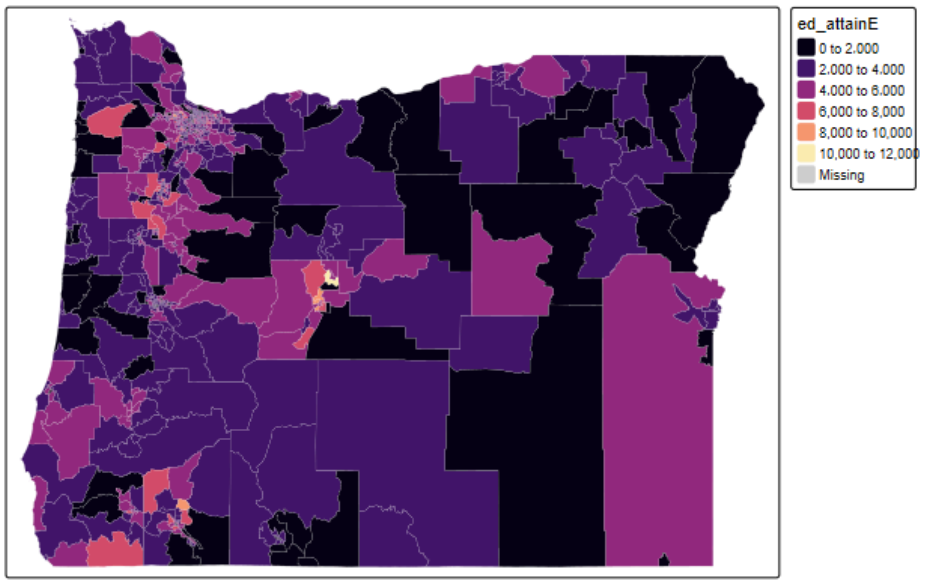
Facet

```
tm_shape(census_vals) +
  tm_polygons("estimate") +
  tm_facets("variable") +
  tm_layout(legend.outside = TRUE)
```



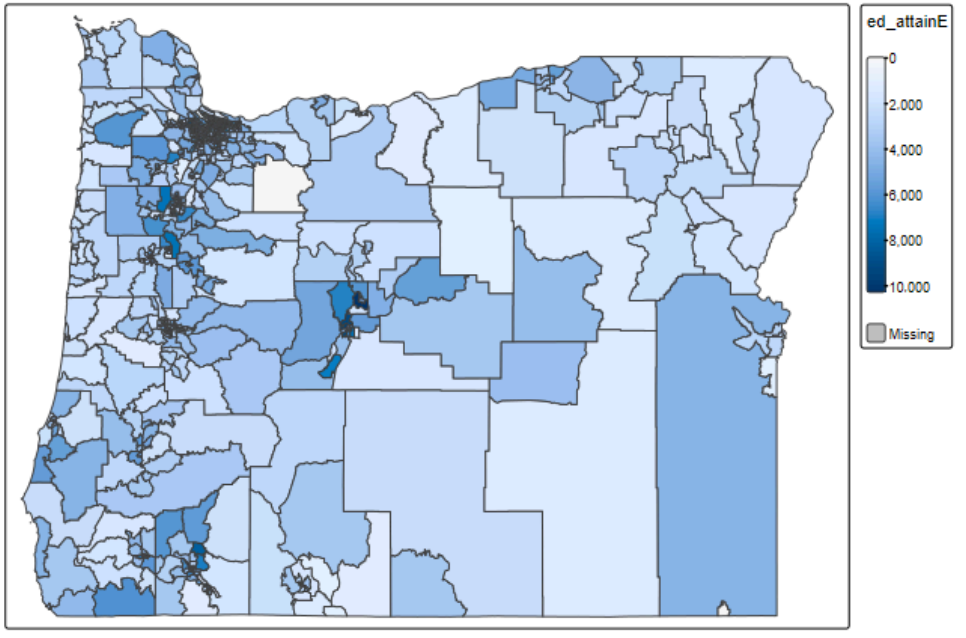
Change colors

```
tm_shape(wider) +
  tm_polygons("ed_attainE",
              palette = "magma",
              border.col = "gray90",
              lwd = 0.1) +
  tm_layout(legend.outside = TRUE)
```



Continuous legend

```
tm_shape(wider) +
  tm_polygons("ed_attainE",
              style = "cont") +
  tm_layout(legend.outside = TRUE)
```



Add text

- First, let's get data at the county level, instead of census tract level

```
cnty <- get_acs(
  geography = "county",
  state = "OR",
  variables = c(ed_attain = "B15003_001"),
  year = 2019,
  geometry = TRUE
)
```

```
##
  |
  |
  |
  |==============
  |
  |================
  |
  |==================
  |
  |====================
  |
```

```
cnty
```

```
## Simple feature collection with 36 features and 5 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -124.5662 ymin: 41.99179 xmax: -116.4635 ymax: 46.29204
## Geodetic CRS:  NAD83
## First 10 features:
##    GEOID                       NAME  variable estimate moe
## 1  41017 Deschutes County, Oregon ed_attain   135615 192 MULTIPOLYGON (((-122.0
## 2  41003    Benton County, Oregon ed_attain    55359 143 MULTIPOLYGON (((-123.8
## 3  41015     Curry County, Oregon ed_attain    18304 139 MULTIPOLYGON (((-124.3
## 4  41061     Union County, Oregon ed_attain    17539  78 MULTIPOLYGON (((-118.6
## 5  41055   Sherman County, Oregon ed_attain     1251  84 MULTIPOLYGON (((-121.0
## 6  41051 Multnomah County, Oregon ed_attain   587290 134 MULTIPOLYGON (((-122.9
## 7  41007   Clatsop County, Oregon ed_attain    28475 165 MULTIPOLYGON (((-123.5
## 8  41033 Josephine County, Oregon ed_attain    63802 148 MULTIPOLYGON (((-124.0
## 9  41031 Jefferson County, Oregon ed_attain    16197  24 MULTIPOLYGON (((-121.8
## 10 41039      Lane County, Oregon ed_attain   256373 147 MULTIPOLYGON (((-124.1
```

Estimate polygon centroid

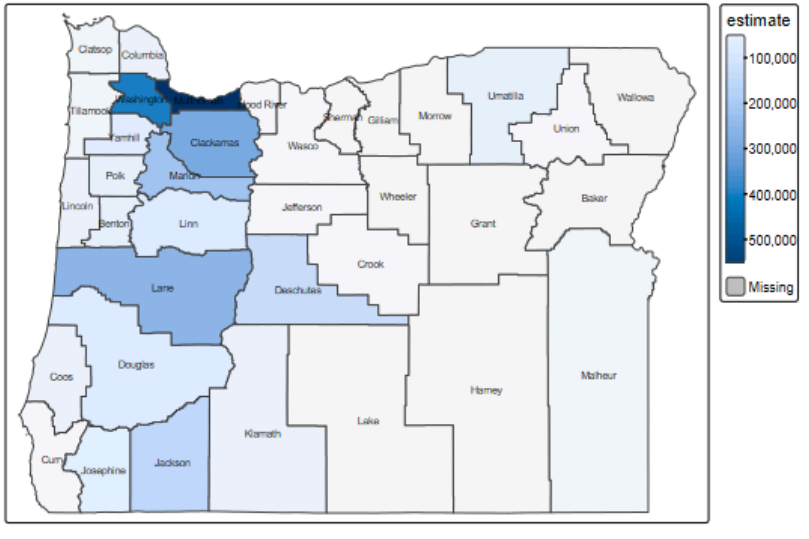
```
centroids <- st_centroid(cnty)
```

Extract just county name

```
centroids <- centroids %>%
  mutate(county = str_replace_all(NAME, " County, Oregon", ""))
```

Plot

```
tm_shape(cnty) +
  tm_polygons("estimate",
              style = "cont") +
tm_shape(centroids) +
  tm_text("county", size = 0.5) +
  tm_layout(legend.outside = TRUE)
```



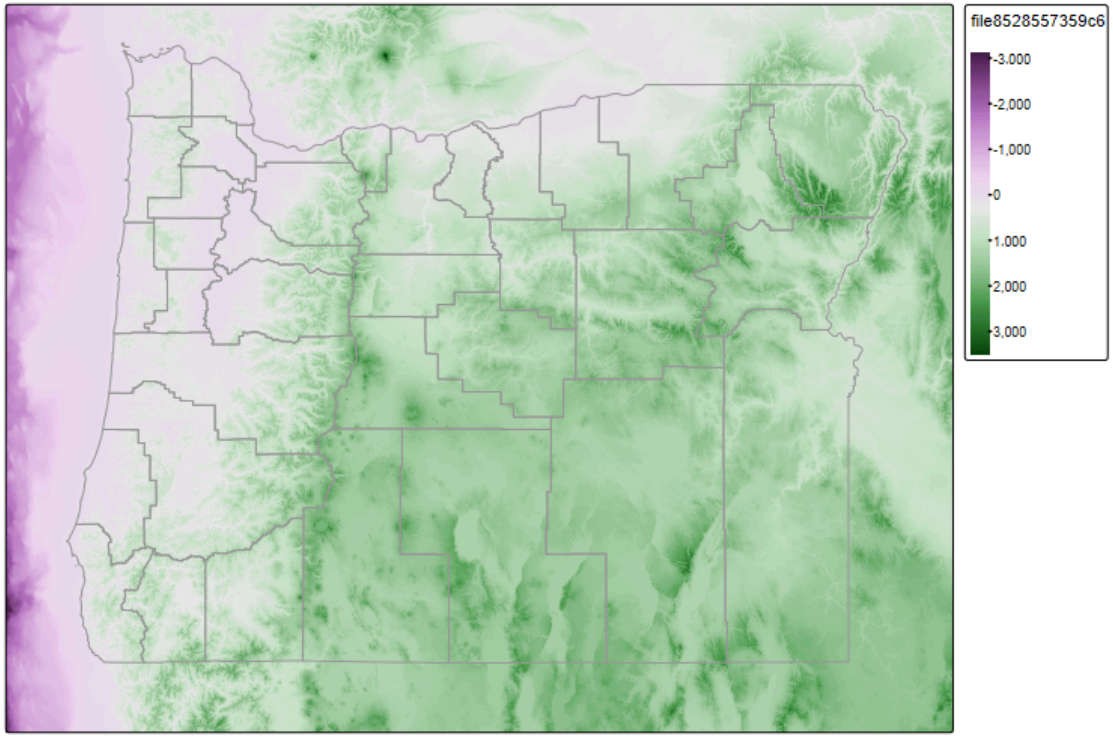
Doesn't work for me on the slides. Not sure why, but should work for you locally.

Add raster elevation data

```
states <- get_acs(
  "state",
  variables = c(ed_attain = "B15003_001"),
  year = 2019,
  geometry = TRUE
)

or <- filter(states, NAME == "Oregon")

# convert to spatial data frame
#sp <- as(or, "Spatial")

# use elevatr library to pull data
library(elevatr)
or_elev <- get_elev_raster(or, z = 9)
lane_elev <- get_elev_raster(or, z = 9)
```

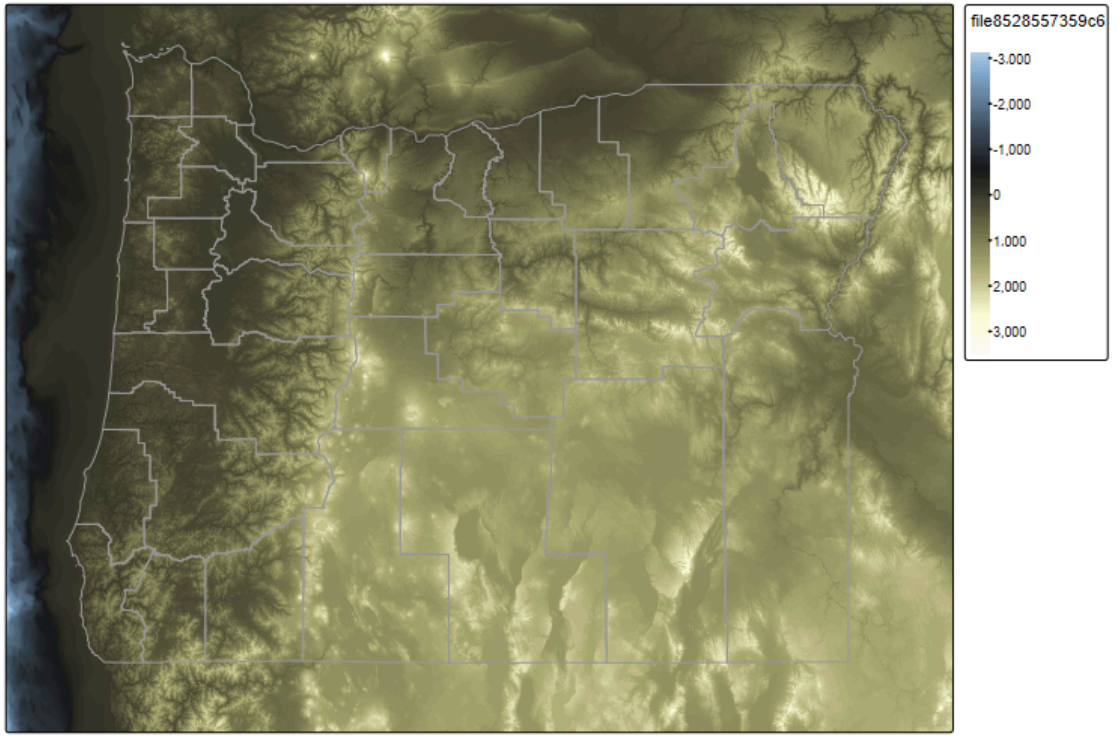
Plot

```
tm_shape(or_elev) +
  tm_raster(midpoint = NA,
            style = "cont") +
  tm_layout(legend.outside = TRUE) +
tm_shape(cnty) +
  tm_borders(col = "gray60")
```

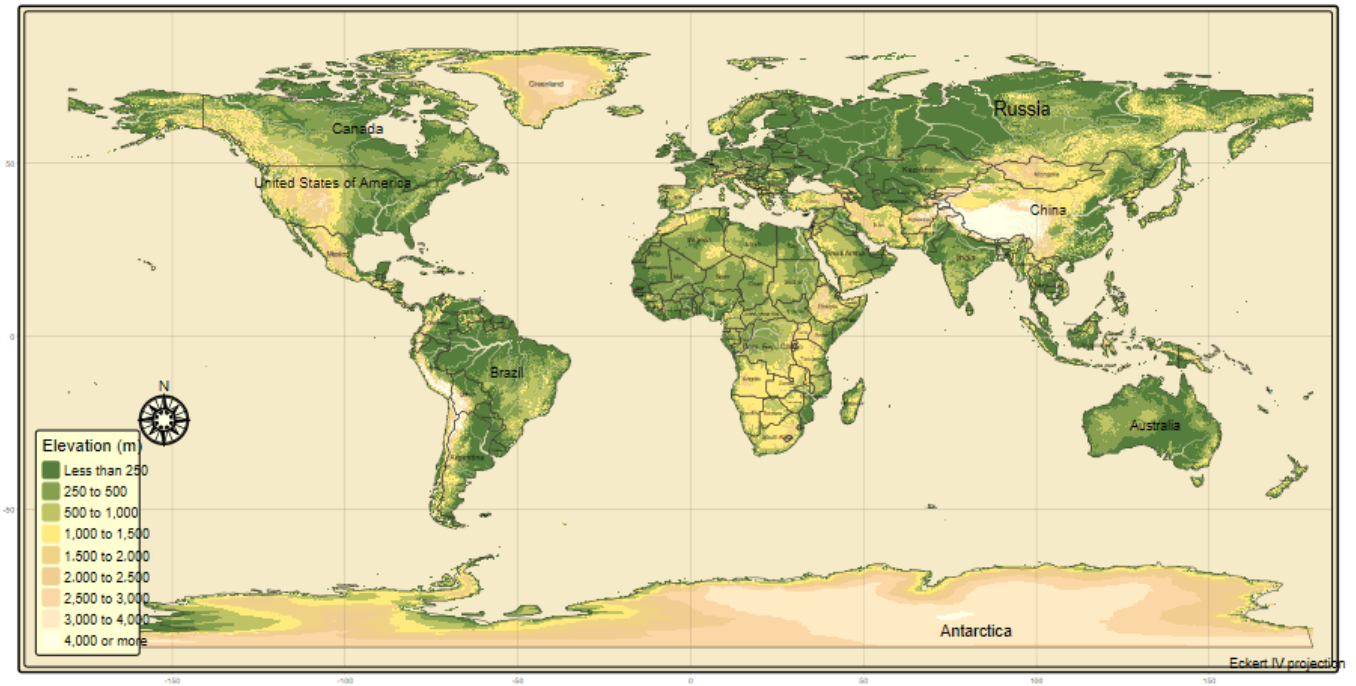


Add custom palette

```
tm_shape(or_elev) +
  tm_raster(midpoint = NA,
            style = "cont",
            palette = c("#E2FCFF", "#83A9CE", "#485C6E",
                        "#181818", "#5C5B3E", "#AAA971",
                        "#FCFCD3", "#ffffff")) +
  tm_layout(legend.outside = TRUE) +
tm_shape(cnty) +
  tm_borders(col = "gray60")
```


file8528557359c6
-3,000
-2,000
-1,000
0
1,000
2,000
3,000

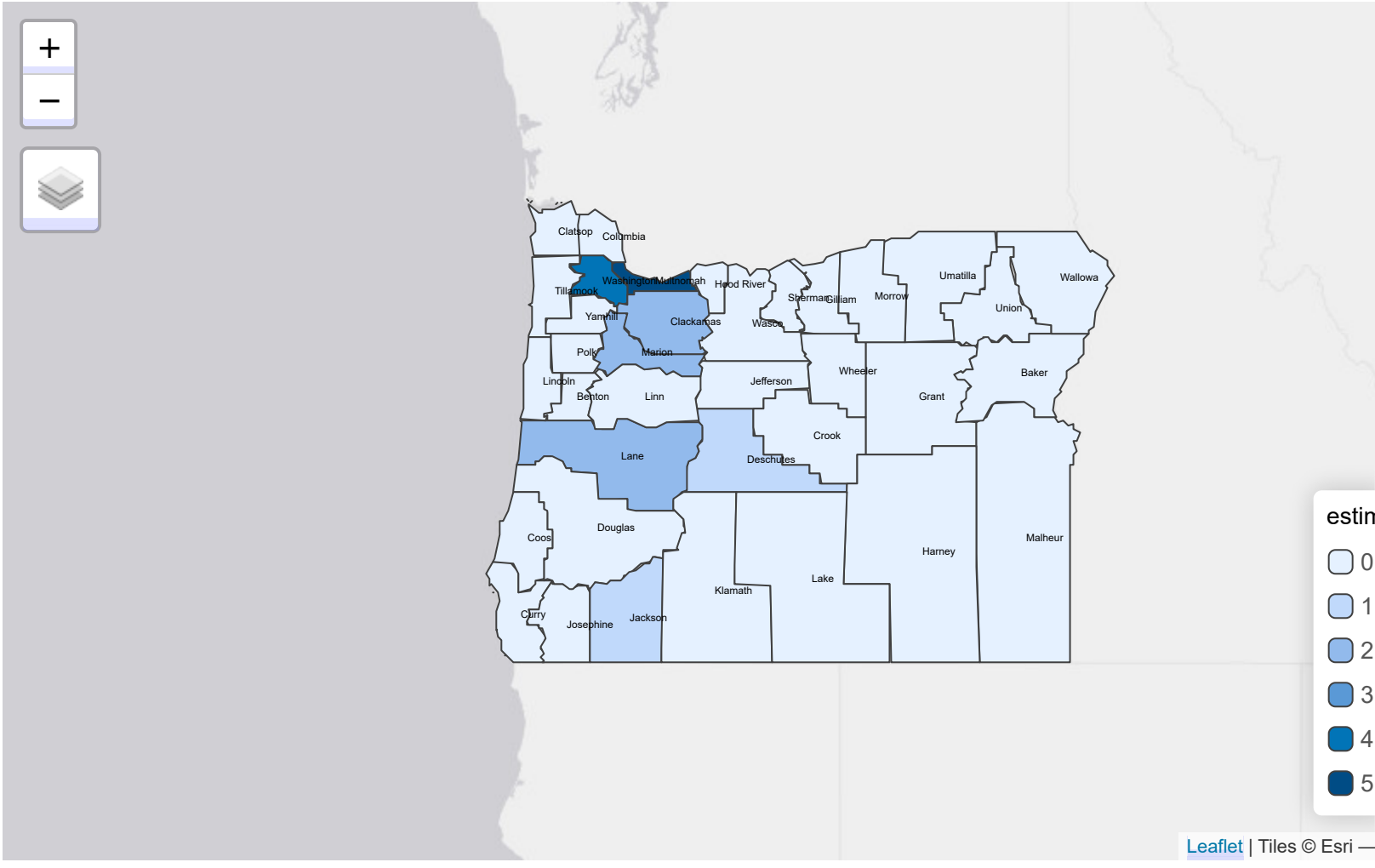
You can do some amazing things!



Create interactive maps

Just change run `tmap_mode("view)` then run the same code as before

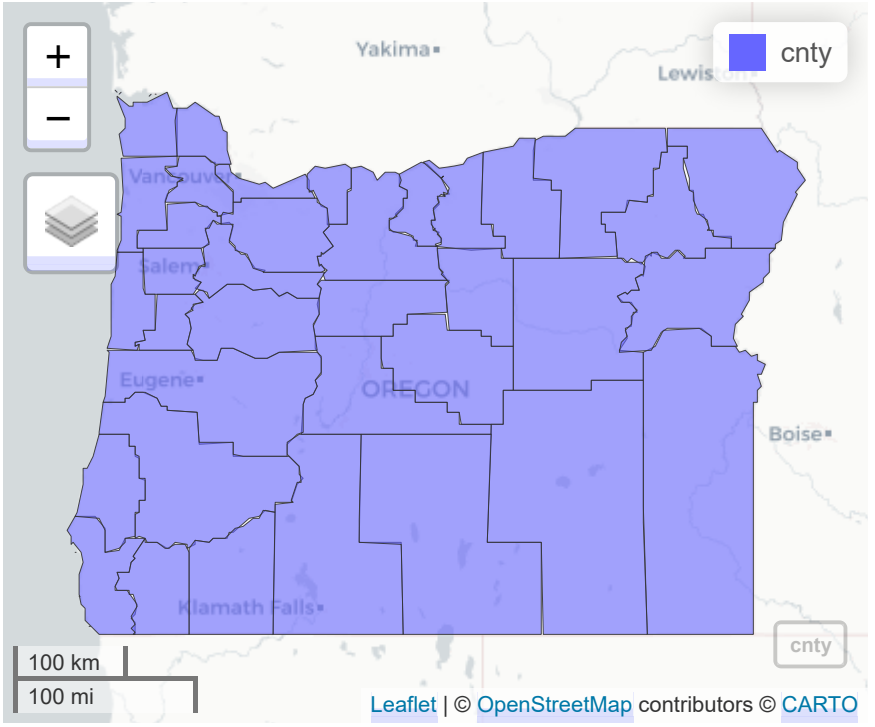
```
tmap_mode("view")

tm_shape(cnty) +
  tm_polygons("estimate") +
tm_shape(centroids) +
  tm_text("county", size = 0.5)
```


Clatsop
Columbia
Tillamook
Washington
Multnomah
Hood River
Sherman
Gilliam
Morrow
Umatilla
Wallowa
Union
Yamhill
Clackamas
Wasco
Polk
Marion
Jefferson
Wheeler
Baker
Lincoln
Benton
Linn
Grant
Crook
Lane
Deschutes
Douglas
Coos
Harney
Malheur
Lake
Klamath
Curry
Josephine
Jackson
estim
0
1
2
3
4
5
Leaflet | Tiles © Esri —

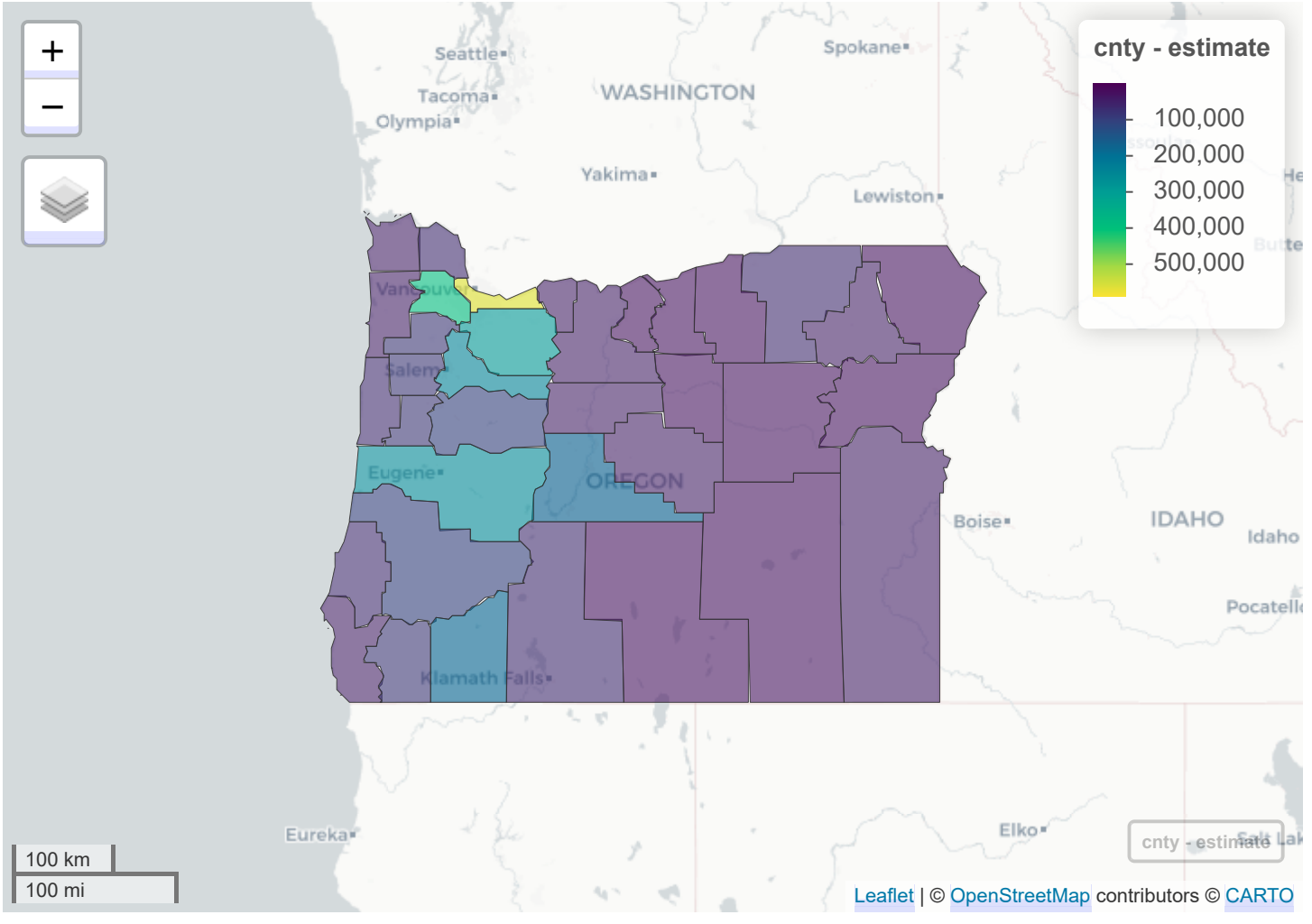
mapview

- Really quick easy interactive maps

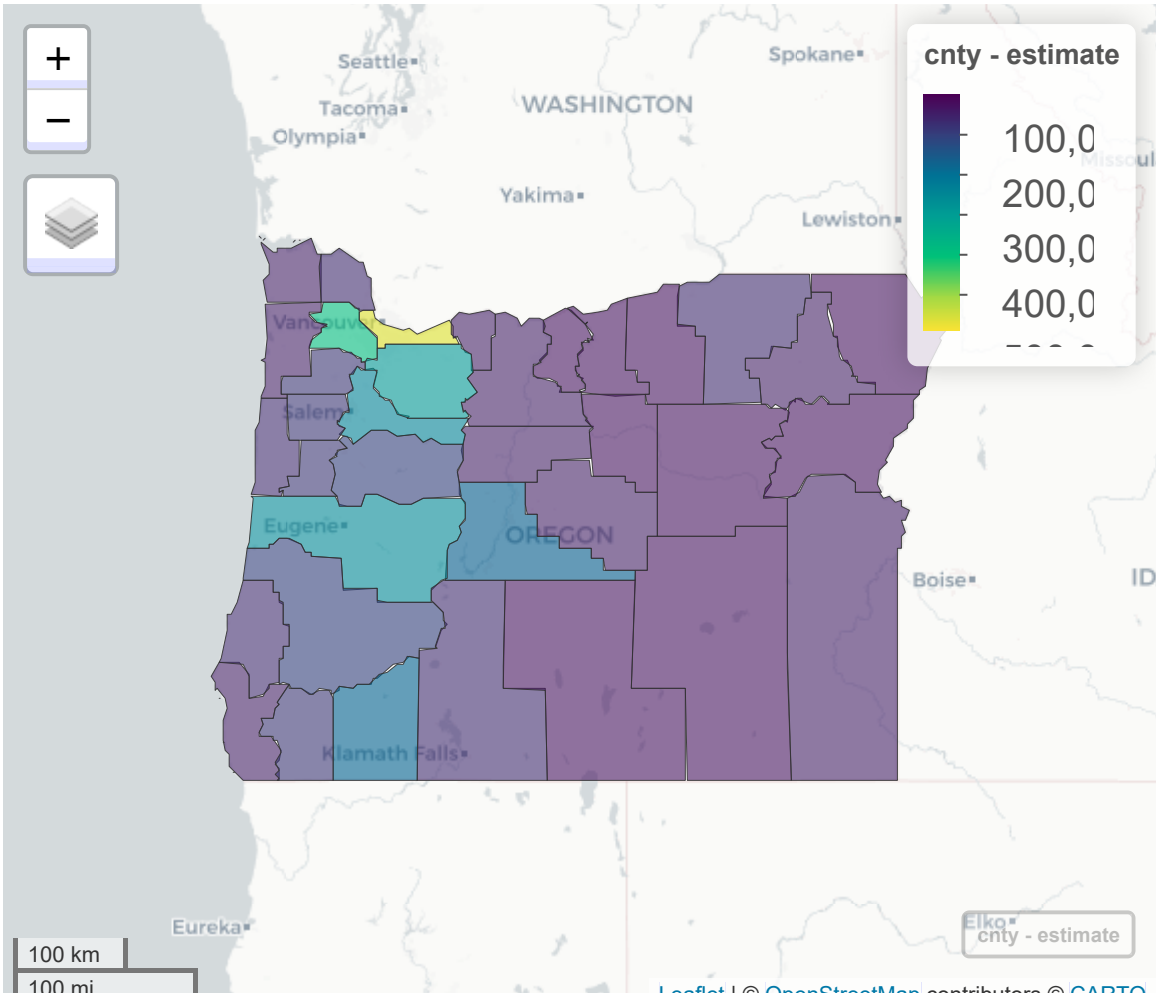
```
library(mapview)
mapview(cnty)
```



```
mapview(cnty, zcol = "estimate")
```



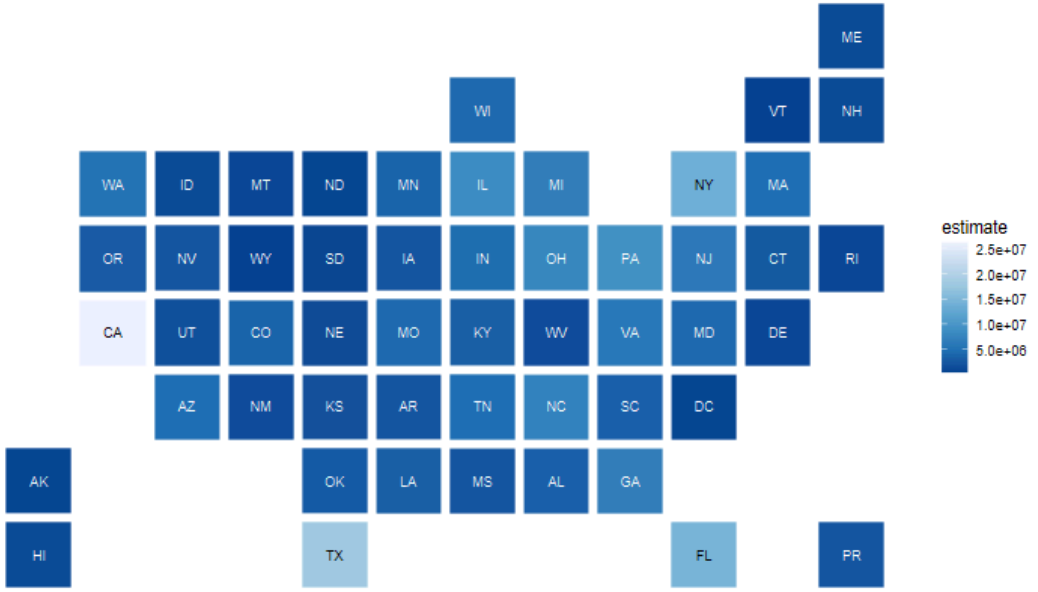
```
mapview(cnty,
        zcol = "estimate",
        popup = leafpop::popupTable(cnty,
                                    zcol = c("NAME", "estimate")))
```



A few other things of note

statebins

```
library(statebins)
statebins(states,
          state_col = "NAME",
          value_col = "estimate") +
  theme_void()
```



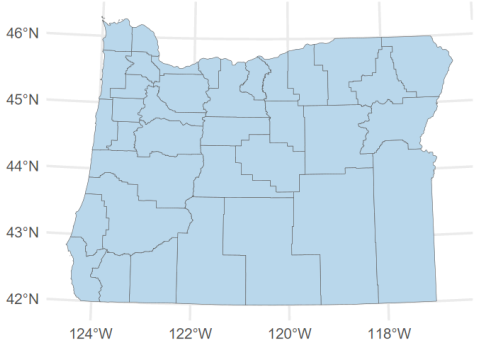
Cartograms

```
library(cartogram)
or_county_pop <- get_acs(
  geography = "county",
  state = "OR",
  variables = "B01001E_001",
  year = 2018,
  geometry = TRUE
)
```

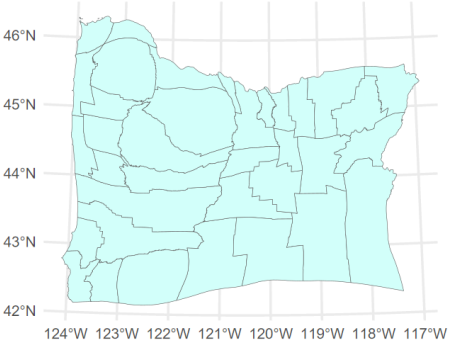
```
##
  |
  |
  |
  |================
  |
  |==================
  |
  |====================
  |
  |======================
  |
  |========================
  |
  |==========================
```

Compare

```
ggplot(or_county_pop) +
  geom_sf(fill = "#BCD8EB")
```




```
ggplot(carto_counties) +
  geom_sf(fill = "#D5FFFA")
```

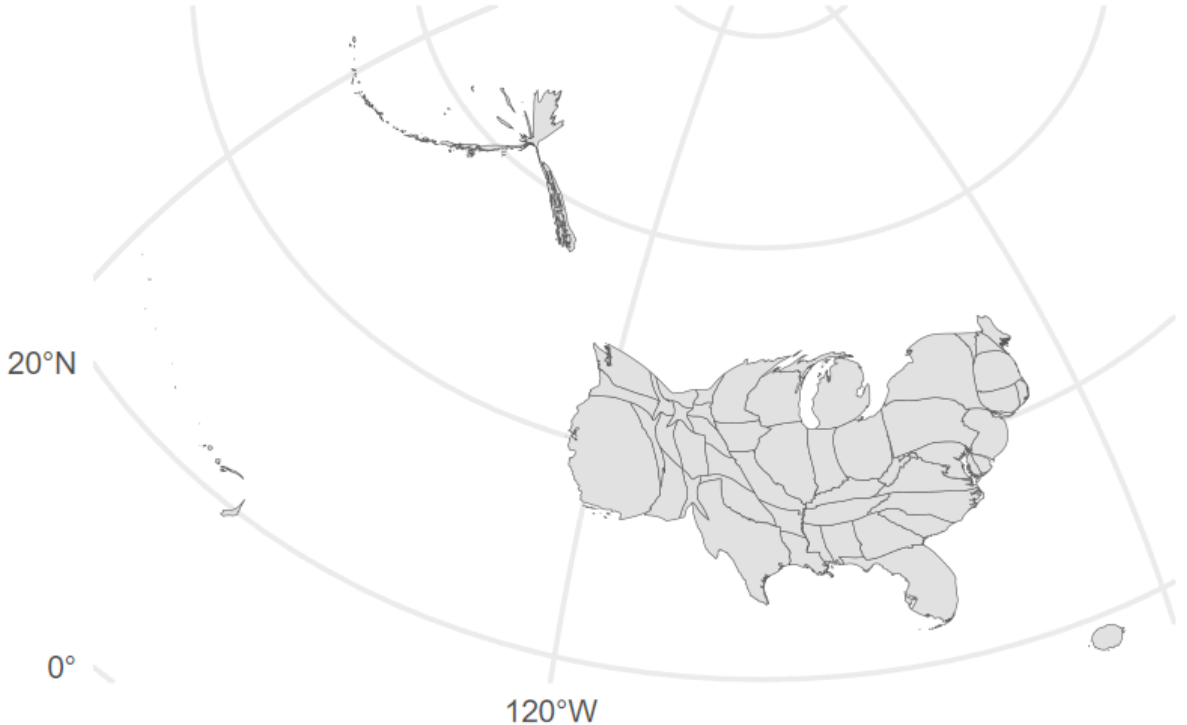


State

```
state_pop <- get_acs(
  geography = "state",
  variables = "B00001_001",
  year = 2018,
  geometry = TRUE
)

# Set projection
state_pop <- st_transform(state_pop, crs = 2163)

# found  the CRS here: https://epsg.io/transform#s_srs=3969&t_srs
carto_states <- cartogram_cont(state_pop, "estimate")
```

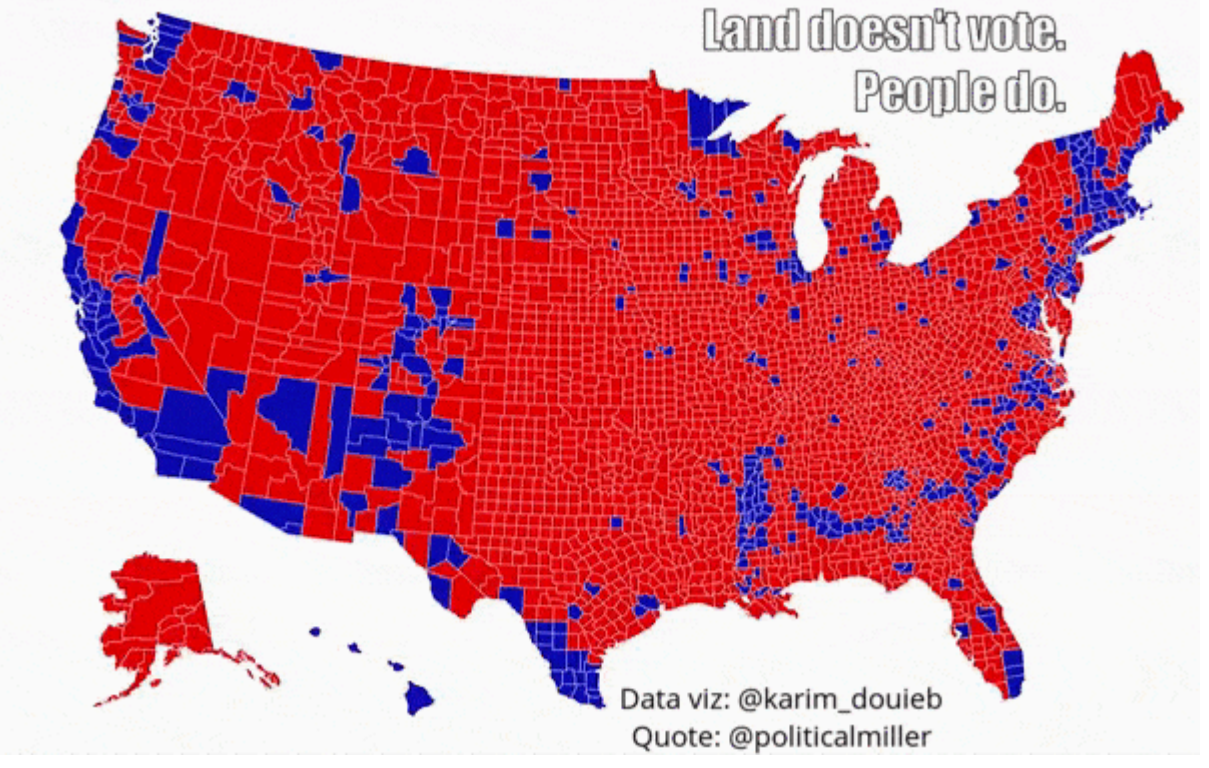
Cartogram of USA by population

```
ggplot(carto_states) +
  geom_sf()
```



Last note

You may or may not like cartograms. Just be careful not to lie with maps.



Loose ends with HTML/Websites

- Can I use Quatro blog instead of a Distill blog?
- Best way to learn more about websites - Fork, clone, mess on your own
- Github pages deployment is the easiest and a good entry point
- But you can use others like Netlify and such for more complicated sites
- Troubleshooting and persistence through that is the key for any skill but especially so for programming!
- So, keep at it!

Presentations for next week - let's quickly look at rubric

Presentations

- Each person/group gets 10 minutes
- Followed by Q & A and a quick 2-minute discussion of plans to finalize and what finishing touches you will add!