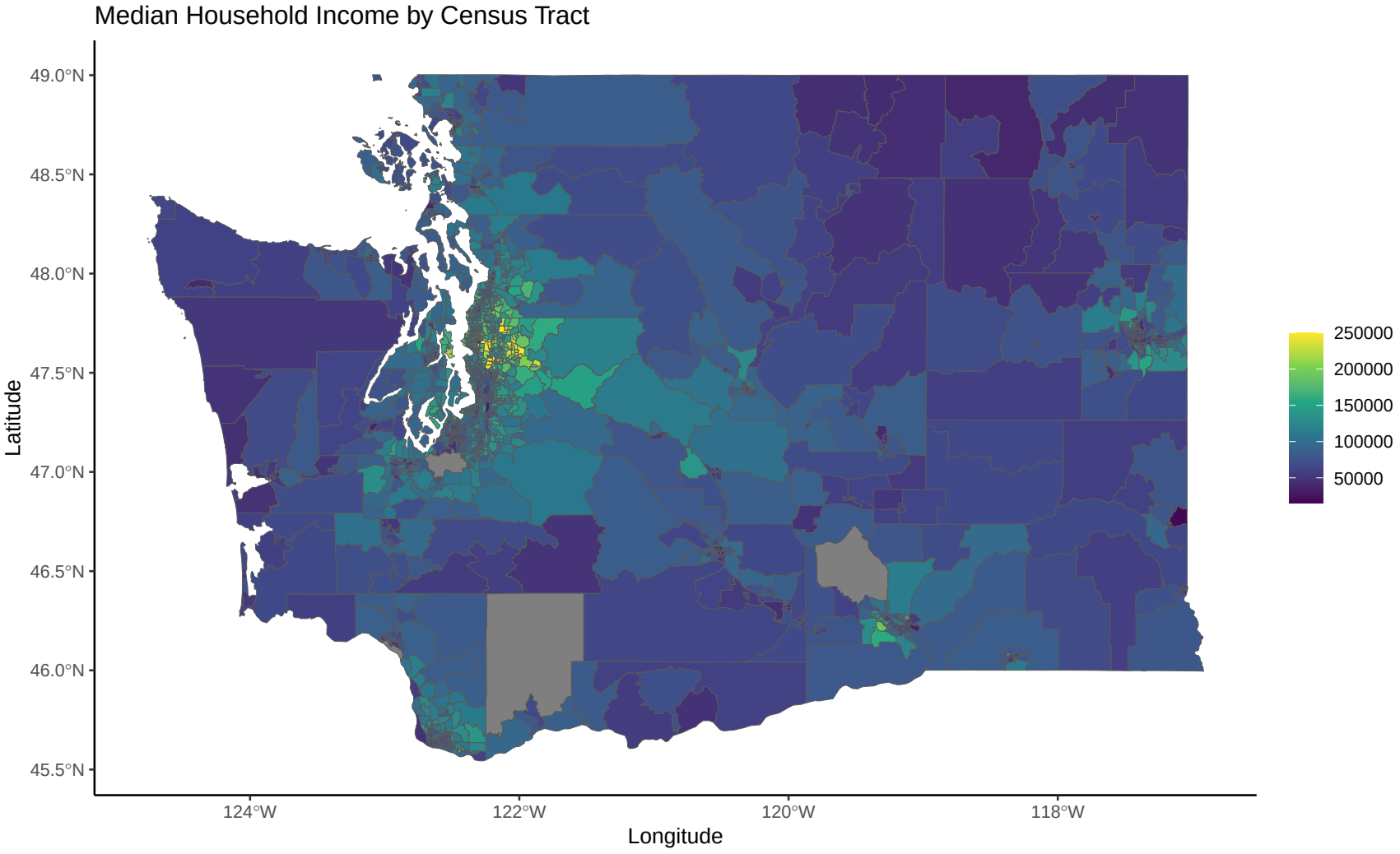
Mapping

Making maps in R

Maps can be tricky in R! There are many packages to choose from.

- `usmap` is compatible with ggplot
- `maps` is "Base R"
- Some require API keys (e.g., `ggmap`, `tidycensus`)
- Some are interactive (e.g., `leaflet`)
- Some provide utilities (e.g., `tigris`)
- Some deal with raster data (e.g., `terra`)

What does a map in R look like?



Data formats - boundary data

```
library(tidyverse)
head(map_data("county"))
```

```
       long      lat group order  region subregion
1 -86.50517 32.34920     1     1 alabama   autauga
2 -86.53382 32.35493     1     2 alabama   autauga
3 -86.54527 32.36639     1     3 alabama   autauga
4 -86.55673 32.37785     1     4 alabama   autauga
5 -86.57966 32.38357     1     5 alabama   autauga
6 -86.59111 32.37785     1     6 alabama   autauga
```

Data formats - boundary data

- `long`: Longitude (x-coordinate)
- `lat`: Latitude (y-coordinate)
- `group`: Identifies unique polygons (each county may have multiple polygons if it contains islands or complex borders).
- `order`: Sequence in which points should be connected to form the boundary (polygons).
- `region`: State name (e.g., "alabama").
- `subregion`: County name within the state (e.g., "autauga").

Data formats - sf data

```
library(usmapdata)
head(us_map("county"))
```

```
Simple feature collection with 6 features and 4 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -2590847 ymin: -2608148 xmax: -1298969 ymax: -2034041
Projected CRS: NAD27 / US National Atlas Equal Area
# A tibble: 6 x 5
  fips  abbr  full   county                                                geo
  <chr> <chr> <chr>  <chr>                                                   <MULTIPOLYGON [m]
1 02013 AK    Alaska Aleutians East Borough     (((-1762715 -2477334, -1761280
2 02016 AK    Alaska Aleutians West Census Area (((-2396847 -2547721, -2393297
3 02020 AK    Alaska Anchorage Municipality     (((-1517576 -2089908, -1517636
4 02050 AK    Alaska Bethel Census Area         (((-1905141 -2137046, -1900900
5 02060 AK    Alaska Bristol Bay Borough        (((-1685825 -2253496, -1684030
6 02063 AK    Alaska Chugach Census Area        (((-1476669 -2101298, -1469831
```

Data formats - sf data

This data is "Simple Feature" (sf) data used for spatial analysis.

These objects store geometric shapes (like points, lines, or polygons) along with associated attributes (metadata).

the `geom` column is a `MULTIPOLYGON` — a geometry type representing complex shapes, which may consist of multiple polygons (e.g., islands or non-contiguous regions).

Federal Information Processing System (FIPS) Codes for States and Counties are numbers which uniquely identify geographic areas. See this codebook.

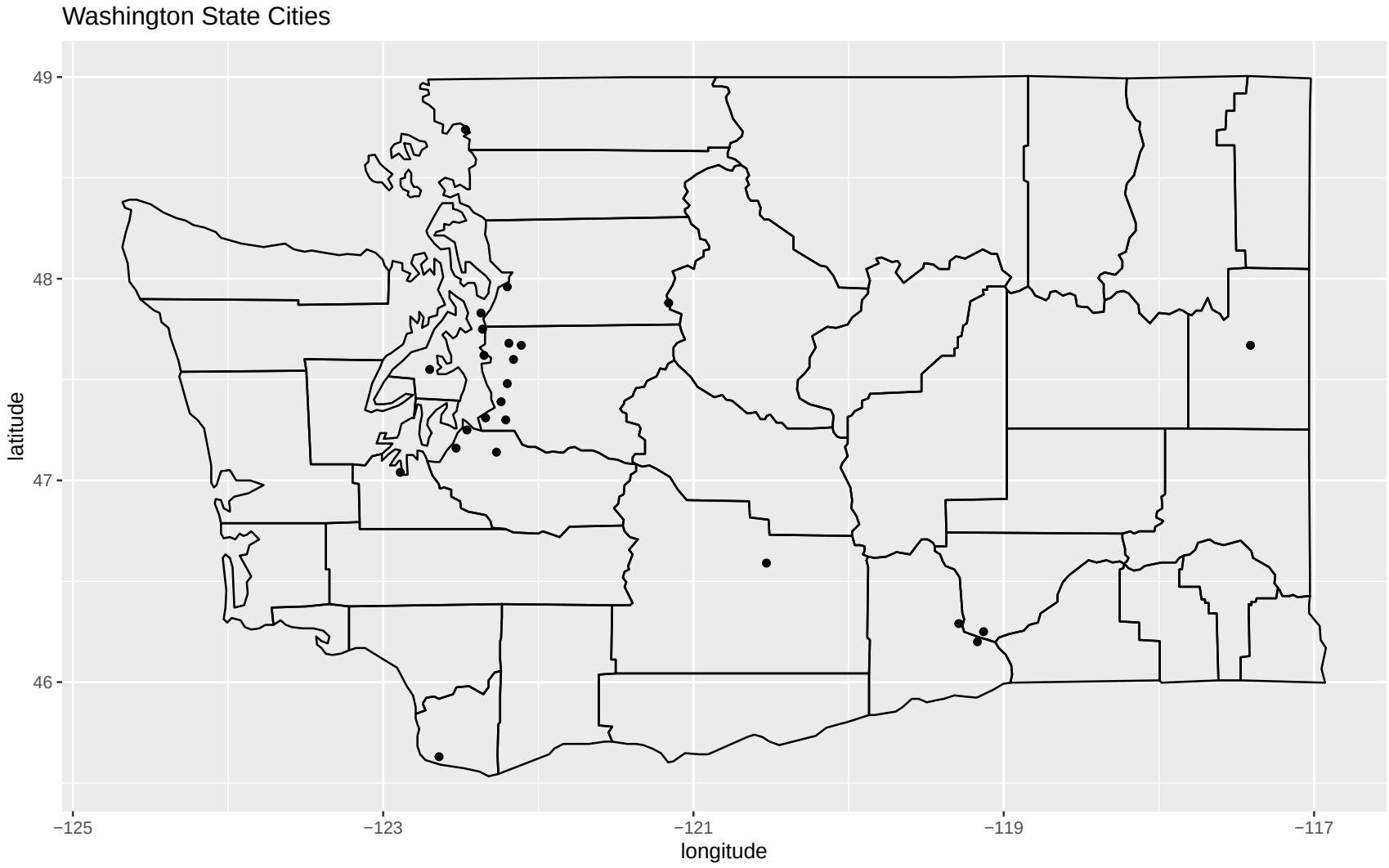
ggplot has spatial functions

geom_polygon() works with boundary data

Let's plot county outlines and major cities.

```
library(tidyverse) # `map_data()` from ggplot2
library(maps) # `us.cities` data

wa_county <- map_data("county") |> filter(region == "washington")
wa_cities <- us.cities |> filter(country.etc == "WA")

plot_1 <-
  ggplot() +
  geom_polygon(data = wa_county, aes(x = long, y = lat, group = group),
               color = "black", fill = NA) +
  geom_point(data = wa_cities, aes(x = long, y = lat)) +
  labs(title = "Washington State Cities", x = "longitude", y = "latitude") +
  coord_fixed(1.3)
```

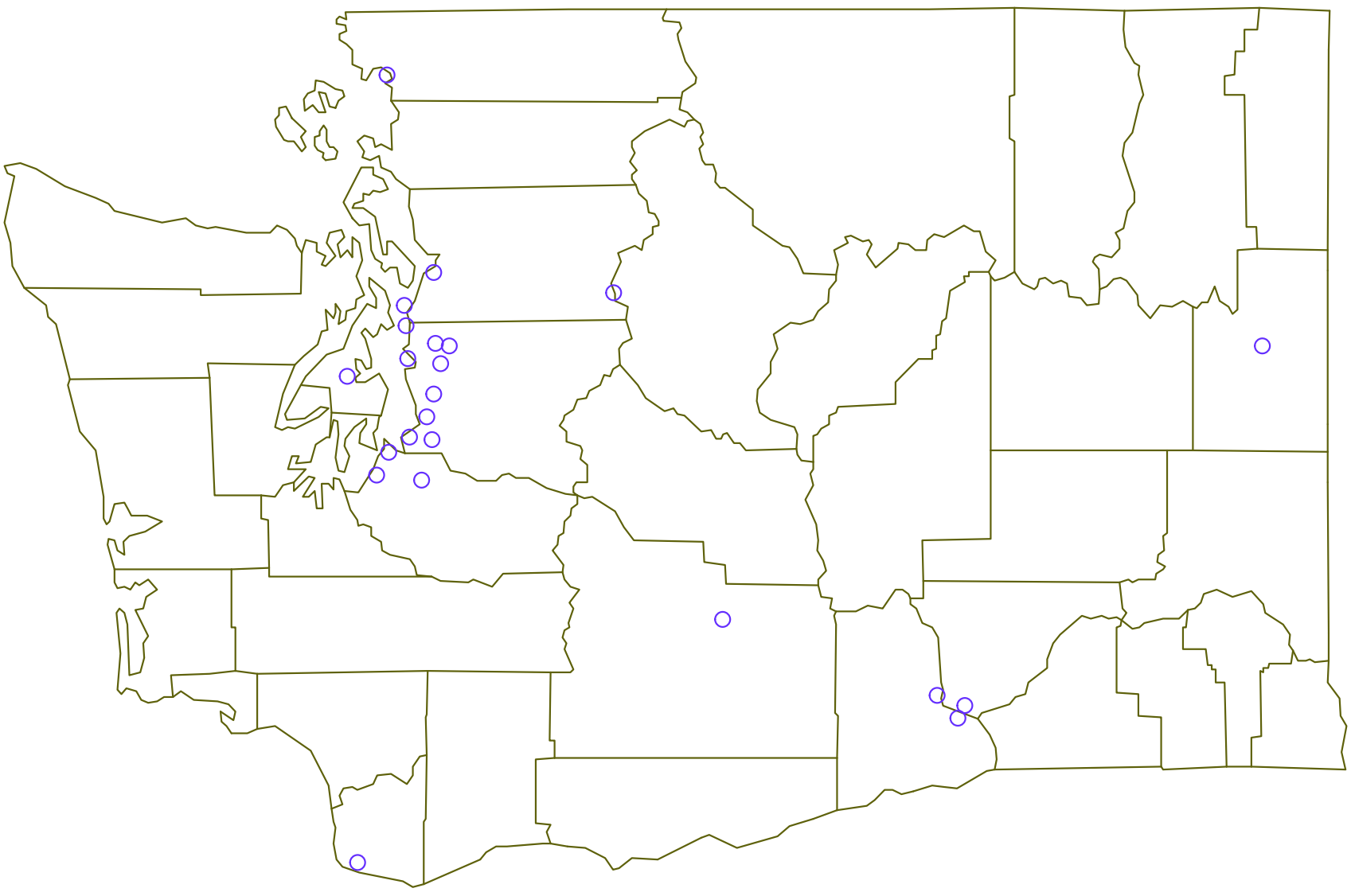
ggplot has spatial functions



maps package is more similar to Base R

```
library(maps) # `map` and `map.cities` functions and `us.cities` data

map('county', region = 'washington', col = "#5E610B")
map.cities(us.cities, country="WA", col = "#642EFE", cex = 2)
title(main = "Washington State Cities")
```

maps package is more similar to Base R

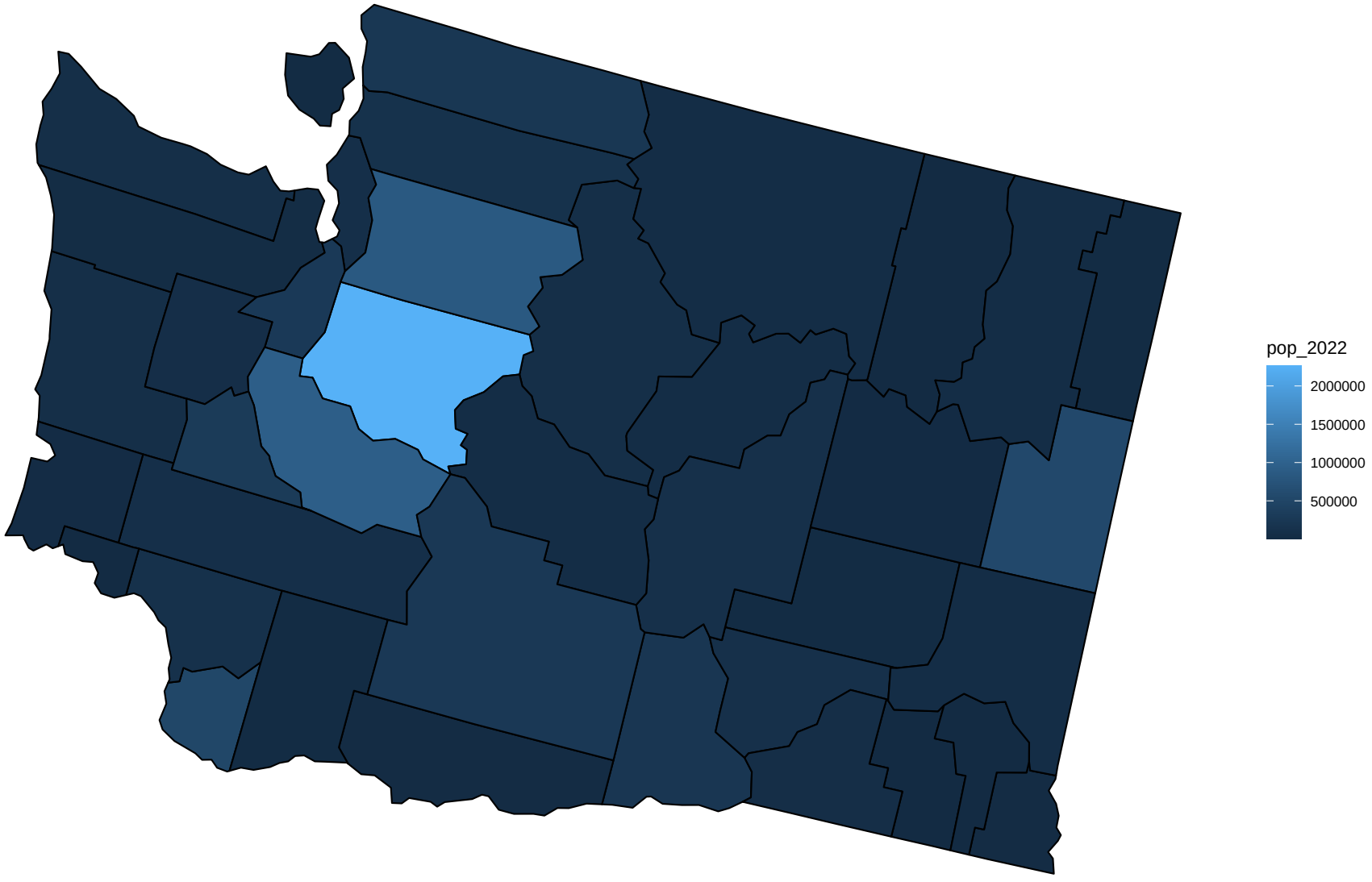


usmap is compatible with ggplot

Let's fill each county based on its population.

```
library(tidyverse)
library(usmap) # `countypop` data and the `plot_usmap()` function

wa_dat <- countypop |> filter(abbr == "WA")

plot_3 <-
  plot_usmap(data = wa_dat, values = "pop_2022", include = c("WA")) +
  scale_fill_continuous() +
  theme(legend.position = "right")
```

usmap is compatible with ggplot



It can get complicated! ggplot fill by county

Sometimes a lot of cleanup is needed to join boundary data with attributes of interest!

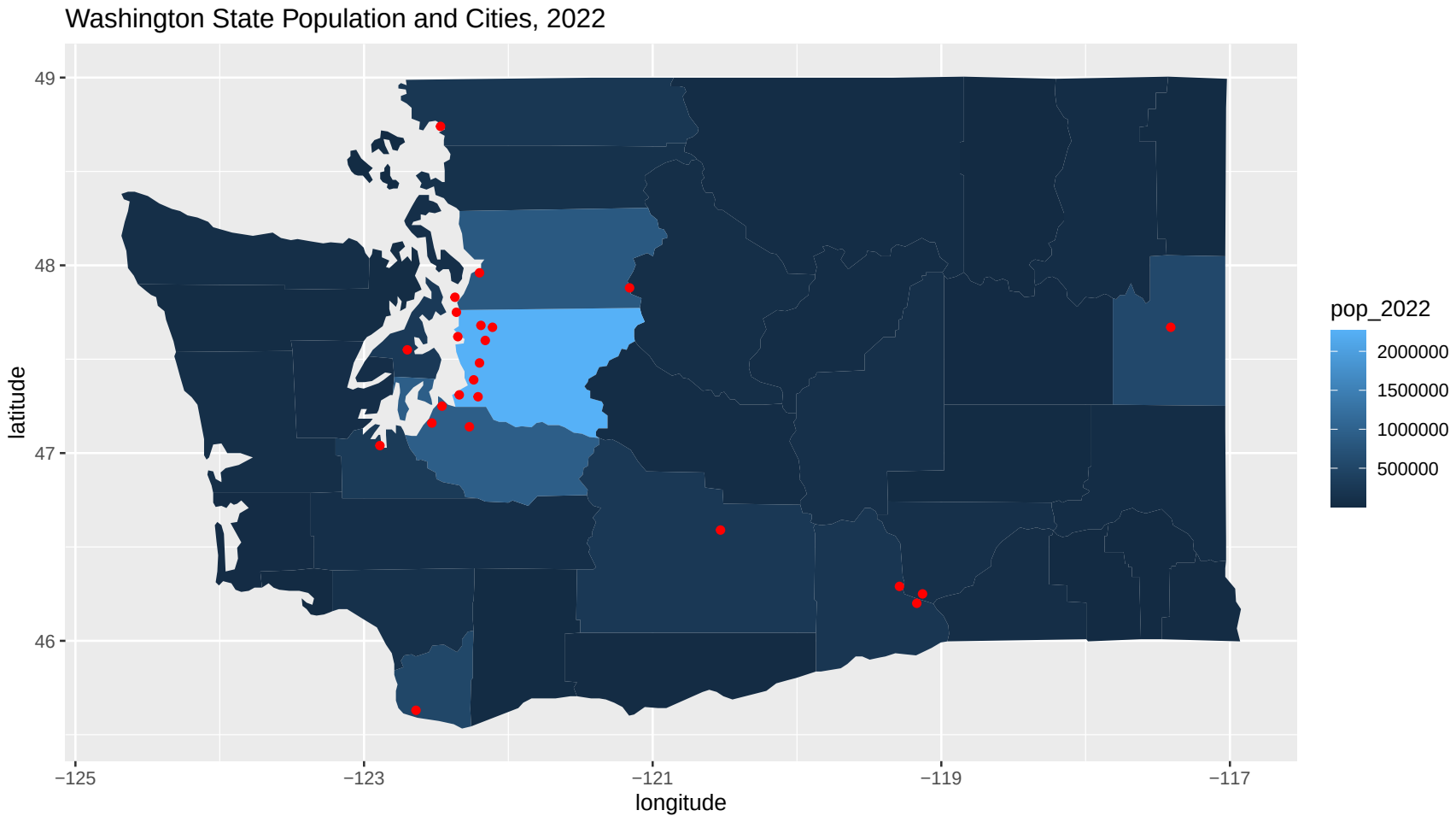
```
library(tidyverse)
library(usmap) # `countypop` data
library(maps) # `us.cities` data

# Get county boundaries
wa_county <- map_data("county") |> filter(region == "washington")

# Get county-level ("subregion") population
wa_dat <- countypop |> filter(abbr == "WA") |>
  mutate(subregion = tolower(str_remove(county, " County"))) |>
  group_by(subregion) |> summarize(pop_2022 = sum(pop_2022))

# Combine the data
wa_complete <- wa_county |> inner_join(wa_dat)

# Get WA cities and their coordinates
wa_cities <- us.cities |> filter(country.etc == "WA")
```

It can get complicated! ggplot fill by county

Sometimes a lot of cleanup is needed to join boundary data with attributes of interest!

```
# Step 2: create the plot
plot_4 <-
  ggplot() +
  geom_polygon(data = wa_complete,
               aes(x = long, y = lat, group = group, fill = pop_2022)) +
  geom_point(data = wa_cities, aes(x = long, y = lat), color = "red") +
  labs(
    title = "Washington State Population and Cities, 2022",
    x = "longitude", y = "latitude") +
  coord_fixed(1.3)
```

It can get complicated! `ggplot` fill by county



tidycensus is helpful for tract level

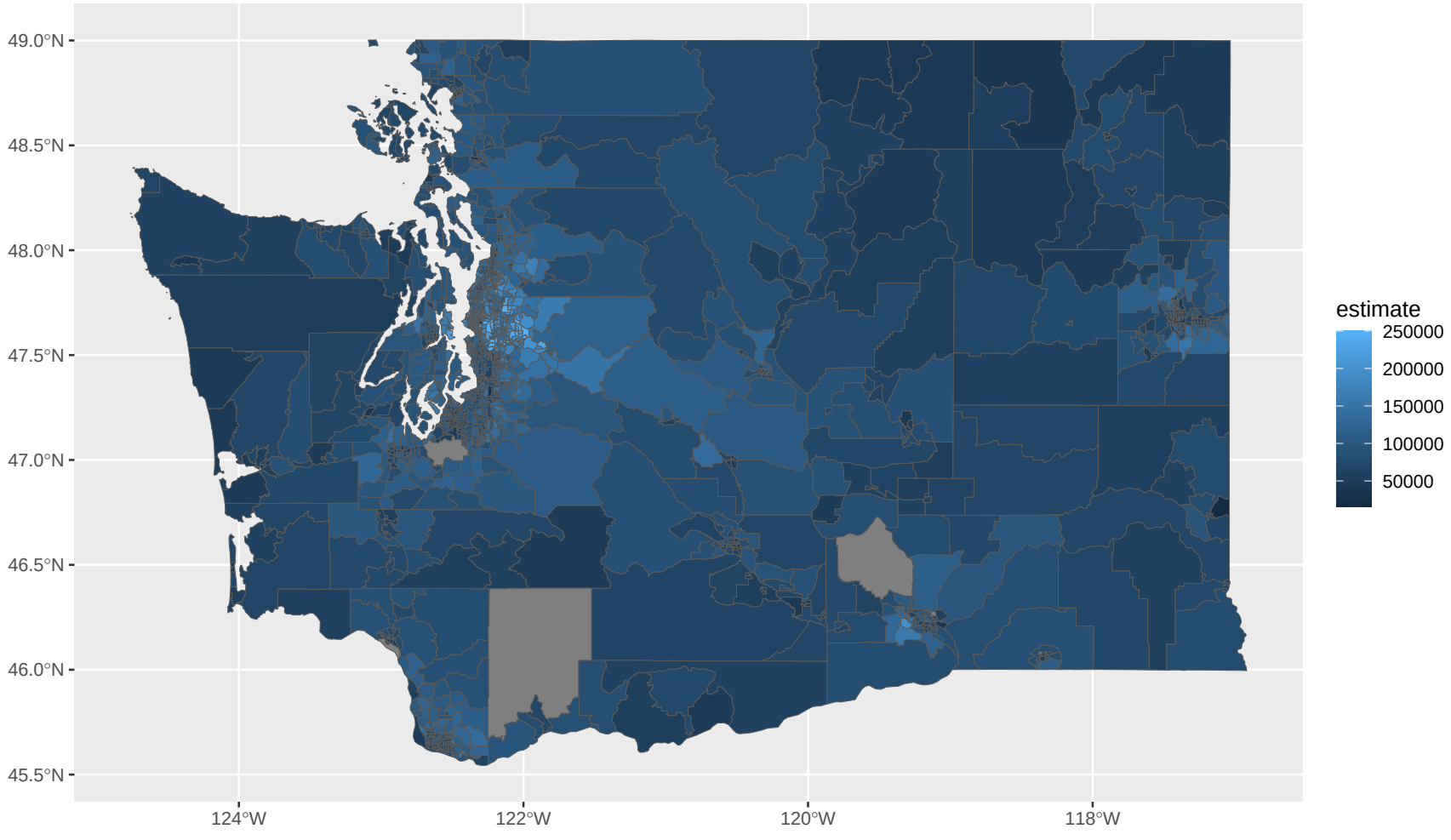
Use `geom_sf()` function with SF data.

Let's fill each census tract by median household income.

```
library(tidyverse) # `geom_sf()` from ggplot2
library(tidycensus) # `get_acs()` function for American Community Survey data

wa_income <- get_acs(
  geography = "tract",
  variables = "B19013_001", # Median income code
  state = "WA",
  year = 2022,
  geometry = TRUE
)
```

tidycensus is helpful for tract level

```
ggplot(data = wa_income, aes(fill = estimate)) +
  geom_sf()
```



Tips for Mapping in R

1. Know the functions: make sure your data going into plotting functions is similar to
2. Data Structure: Ensure column names match between datasets for join() operations
   - e.g., `subregion` needs to align in both `wa_county` and `wa_dat` to make `wa_complete`.
   - Make sure all datasets (like counties and cities) use the same geographic system, such as longitude-latitude pairs.
3. Clean Data to make life easier
   - Use functions like `tolower()` and `str_remove()` to standardize text (e.g., removing "County").
   - Group and summarize data when plotting aggregates, like population by county.

More resources

https://rspatial.org/index.html (whole course using `terra`!)

https://rfortherestofus.com/2024/06/us-maps

https://ggplot2-book.org/maps

https://walker-data.com/tidycensus/articles/spatial-data.html

https://walker-data.com/census-r/mapping-census-data-with-r.html

https://jtr13.github.io/cc19/different-ways-of-plotting-u-s-map-in-r.html

tigris has many utility functions

tigris gets boundary shapefiles from the US Census Bureau (as sf data).

- zctas(): ZIP code tabulation areas
- voting_districts(): voting districts
- school_districts(): school districts

```
library(tigris)
```

```
To enable caching of data, set `options(tigris_use_cache = TRUE)`
in your R script or .Rprofile.
```

```
zip_wa <- zctas(state = "WA", year = 2010)
```

```
ZCTAs can take several minutes to download.  To cache the data and avoid re-dow
  |
```

```
zip_wa
```

```
Simple feature collection with 598 features and 11 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -124.7428 ymin: 45.54354 xmax: -116.9156 ymax: 49.00249
Geodetic CRS:  NAD83
First 10 features:
   STATEFP10 ZCTA5CE10 GEOID10 CLASSFP10 MTFCC10 FUNCSTAT10    ALAND10 AWATER10
1         53     98822 5398822        B5   G6350          S 1131837710  5582389
2         53     98821 5398821        B5   G6350          S    4754899   198324
3         53     98357 5398357        B5   G6350          S  110004759   462073
4         53     98663 5398663        B5   G6350          S   11134084    70154
```