US-census-Data-Analysis.R

paulbedu-osei

2026-01-01

```r
#loading libraries
library("tidycensus")
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.6
## v forcats   1.0.1     v stringr   1.6.0
## v ggplot2   4.0.1     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.2.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library("ggplot2")
library("tigris")
```

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

```r
library("sf")
```

```
## Linking to GEOS 3.13.0, GDAL 3.8.5, PROJ 9.5.1; sf_use_s2() is TRUE
```

```r
library("dplyr")
library("lwgeom")
```

```
## Linking to liblwgeom 3.0.0beta1 r16016, GEOS 3.13.0, PROJ 9.5.1
##
## Attaching package: ’lwgeom’
##
## The following objects are masked from ’package:sf’:
##
##     st_minimum_bounding_circle, st_perimeter
```

```r
# Define your Census API key and set it with census_api_key()
api_key <- "c1535c768adf3ad53bc76c4d9b013282f18f871a"
census_api_key(api_key, install)
```

```
## To install your API key for use in future sessions, run this function with `install = TRUE`.
```

```
# Check your API key
Sys.getenv("CENSUS_API_KEY")
```

```
## [1] "c1535c768adf3ad53bc76c4d9b013282f18f871a"
```

```
# Obtain and view state median household income from the American Community Survey
state_income <- get_acs(geography = "state", variables = "B19013_001")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```
head(state_income)
```

```
## # A tibble: 6 x 5
##   GEOID NAME       variable   estimate   moe
##   <chr> <chr>      <chr>         <dbl> <dbl>
## 1 01    Alabama    B19013_001    62027   400
## 2 02    Alaska     B19013_001    89336  1374
## 3 04    Arizona    B19013_001    76872   414
## 4 05    Arkansas   B19013_001    58773   503
## 5 06    California B19013_001    96334   298
## 6 08    Colorado   B19013_001    92470   483
```

```
# Get an ACS dataset for Census tracts in Texas by setting the state
tx_income <- get_acs(geography = "tract",
                     variables = "B19013_001",
                     state = "TX")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```
# Inspect the dataset
head(tx_income)
```

```
## # A tibble: 6 x 5
##   GEOID       NAME                                          variable estimate   moe
##   <chr>       <chr>                                         <chr>       <dbl> <dbl>
## 1 48001950100 Census Tract 9501; Anderson County; Texas  B19013_~    67344 11901
## 2 48001950401 Census Tract 9504.01; Anderson County; Te~ B19013_~    98750 44634
## 3 48001950402 Census Tract 9504.02; Anderson County; Te~ B19013_~       NA    NA
## 4 48001950500 Census Tract 9505; Anderson County; Texas  B19013_~    59358 13224
## 5 48001950600 Census Tract 9506; Anderson County; Texas  B19013_~    44250 15182
## 6 48001950700 Census Tract 9507; Anderson County; Texas  B19013_~    39375 14430
```

```
# Get an ACS dataset for Census tracts in Travis County, TX
travis_income <- get_acs(geography = "tract",
                         variables = "B19013_001",
                         state = "TX",
                         county = "Travis")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```r
# Inspect the dataset
head(travis_income)
```

```
## # A tibble: 6 x 5
##   GEOID       NAME                                      variable   estimate   moe
##   <chr>       <chr>                                     <chr>         <dbl> <dbl>
## 1 48453000101 Census Tract 1.01; Travis County; Texas B19013_001   123050 48545
## 2 48453000102 Census Tract 1.02; Travis County; Texas B19013_001   155897 46935
## 3 48453000203 Census Tract 2.03; Travis County; Texas B19013_001    95778  7321
## 4 48453000204 Census Tract 2.04; Travis County; Texas B19013_001   120464 19511
## 5 48453000205 Census Tract 2.05; Travis County; Texas B19013_001    78287 24468
## 6 48453000206 Census Tract 2.06; Travis County; Texas B19013_001    95703 20510
```

```r
# Supply custom variable names
travis_income2 <- get_acs(geography = "tract",
                          variables = c(variable = "B19013_001"),
                          state = "TX",
                          county = "Travis")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```r
# Inspect the dataset
head(travis_income2)
```

```
## # A tibble: 6 x 5
##   GEOID       NAME                                      variable estimate   moe
##   <chr>       <chr>                                     <chr>       <dbl> <dbl>
## 1 48453000101 Census Tract 1.01; Travis County; Texas variable   123050 48545
## 2 48453000102 Census Tract 1.02; Travis County; Texas variable   155897 46935
## 3 48453000203 Census Tract 2.03; Travis County; Texas variable    95778  7321
## 4 48453000204 Census Tract 2.04; Travis County; Texas variable   120464 19511
## 5 48453000205 Census Tract 2.05; Travis County; Texas variable    78287 24468
## 6 48453000206 Census Tract 2.06; Travis County; Texas variable    95703 20510
```

```r
# Return county data in wide format
or_wide <- get_acs(geography = "county",
                   state = "OR",
                   variables = c(hhincome = "B19013_001",
                                 medage = "B01002_001"),
                   output = "wide")
```

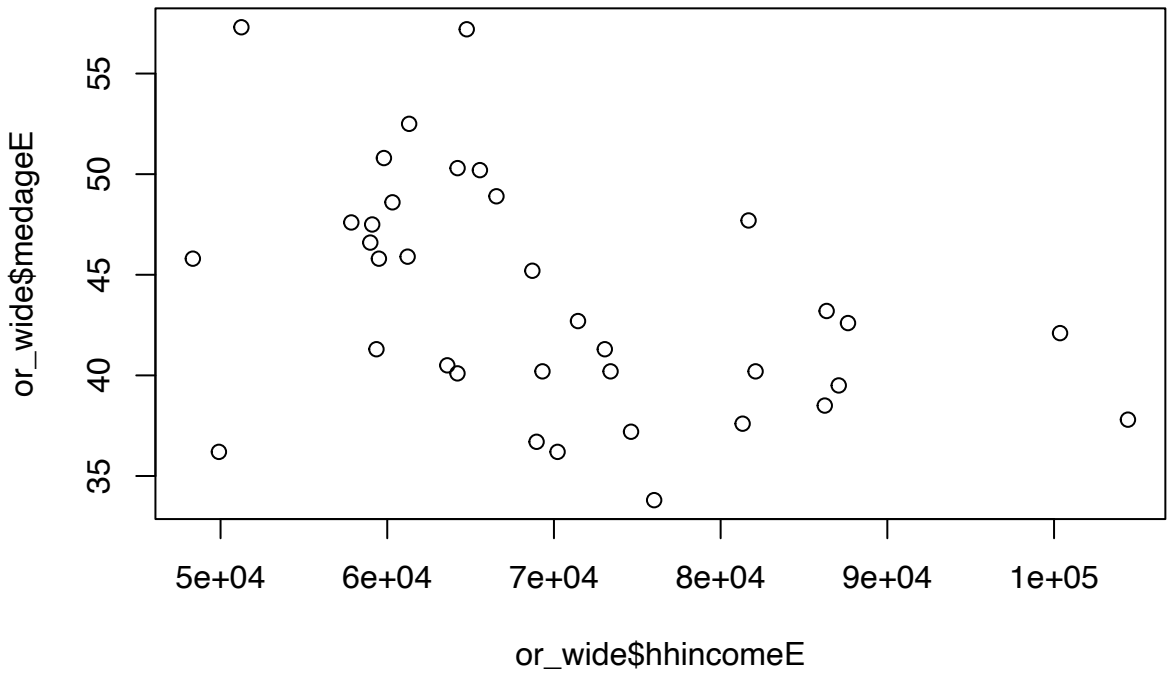
```
## Getting data from the 2019-2023 5-year ACS
```

```r
# Compare output to the tidy format from previous exercises
head(or_wide)
```

```
## # A tibble: 6 x 6
##   GEOID NAME                      hhincomeE hhincomeM medageE medageM
##   <chr> <chr>                         <dbl>     <dbl>   <dbl>   <dbl>
## 1 41001 Baker County, Oregon          57844      4135    47.6     0.8
```

```
## 2 41003 Benton County, Oregon        76011      3407    33.8     0.2
## 3 41005 Clackamas County, Oregon    100360      2211    42.1     0.2
## 4 41007 Clatsop County, Oregon       68705      3596    45.2     0.6
## 5 41009 Columbia County, Oregon      86359      4753    43.2     0.4
## 6 41011 Coos County, Oregon          60313      3431    48.6     0.4
```

```r
# Create a scatterplot
plot(or_wide$hhincomeE, or_wide$medageE)
```




```r
# Get variables from the ACS Data Profile
v16 <- load_variables(year = 2016,
                      dataset = "acs5",
                      cache = TRUE)

# Get variables from the ACS Data Profile
v16p <- load_variables(year = 2016,
                      dataset = "acs5/profile",
                      cache = TRUE)

# Set year and dataset to get variables from the 2000 Census SF3
v00 <- load_variables(year = 2000,
                     dataset = "sf3",
                     cache = TRUE)

# Filter for table B19001
filter(v16, str_detect(name, "B19001"))
```

```
## # A tibble: 170 x 4
##    name        label                                concept             geography
##    <chr>       <chr>                                <chr>               <chr>
##  1 B19001A_001 Estimate!!Total                      HOUSEHOLD INCOME I~ tract
##  2 B19001A_002 Estimate!!Total!!Less than $10,000   HOUSEHOLD INCOME I~ tract
##  3 B19001A_003 Estimate!!Total!!$10,000 to $14,999  HOUSEHOLD INCOME I~ tract
##  4 B19001A_004 Estimate!!Total!!$15,000 to $19,999  HOUSEHOLD INCOME I~ tract
##  5 B19001A_005 Estimate!!Total!!$20,000 to $24,999  HOUSEHOLD INCOME I~ tract
##  6 B19001A_006 Estimate!!Total!!$25,000 to $29,999  HOUSEHOLD INCOME I~ tract
##  7 B19001A_007 Estimate!!Total!!$30,000 to $34,999  HOUSEHOLD INCOME I~ tract
##  8 B19001A_008 Estimate!!Total!!$35,000 to $39,999  HOUSEHOLD INCOME I~ tract
##  9 B19001A_009 Estimate!!Total!!$40,000 to $44,999  HOUSEHOLD INCOME I~ tract
## 10 B19001A_010 Estimate!!Total!!$45,000 to $49,999  HOUSEHOLD INCOME I~ tract
## # i 160 more rows
```

```
# Use public transportation to search for related variables
filter(v16p, str_detect(label, fixed("public transportation",
                                     ignore_case = TRUE)))
```

```
## # A tibble: 2 x 3
##   name       label                                                    concept
##   <chr>      <chr>                                                    <chr>
## 1 DP03_0021  Estimate!!COMMUTING TO WORK!!Workers 16 years and over!!Pu~ SELECT~
## 2 DP03_0021P Percent!!COMMUTING TO WORK!!Workers 16 years and over!!Pub~ SELECT~
```

```
# Access the 1-year ACS  with the survey parameter
ne_income <- get_acs(geography = "state",
                     variables = "B19013_001",
                     survey = "acs1",
                     state = c("ME", "NH", "VT", "MA",
                               "RI", "CT", "NY"))
```

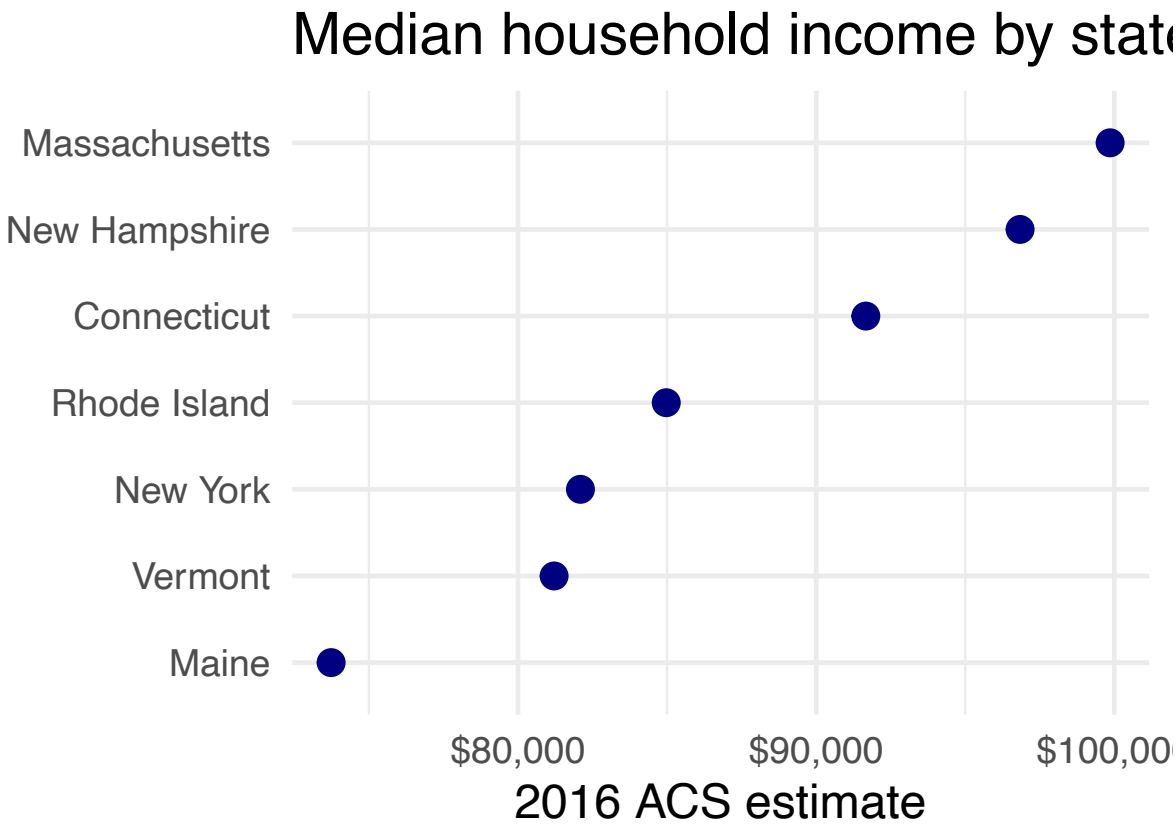
```
## Getting data from the 2023 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
```

```
# Set dot color and size
g_color <- ggplot(ne_income, aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_point(color = "navy", size = 4)

# Format the x-axis labels
g_scale <- g_color +
  scale_x_continuous(labels = scales::dollar) +
  theme_minimal(base_size = 18)

# Label your x-axis, y-axis, and title your chart
g_label <- g_scale +
  labs(x = "2016 ACS estimate",
       y = "",
       title = "Median household income by state")

g_label
```




```
# Download table "B19001"
wa_income <- get_acs(geography = "county",
                     state = "WA",
                     table = "B19001")
```

```
## Getting data from the 2019-2023 5-year ACS
## Loading ACS5 variables for 2023 from table B19001. To cache this dataset for faster access to ACS tab
```

```
# Check out the first few rows of wa_income
head(wa_income)
```

```
## # A tibble: 6 x 5
##   GEOID NAME                      variable   estimate   moe
##   <chr> <chr>                     <chr>         <dbl> <dbl>
## 1 53001 Adams County, Washington B19001_001     6301   166
## 2 53001 Adams County, Washington B19001_002      392   140
## 3 53001 Adams County, Washington B19001_003      233    85
## 4 53001 Adams County, Washington B19001_004      203    95
## 5 53001 Adams County, Washington B19001_005      257   137
## 6 53001 Adams County, Washington B19001_006      474   201
```

```
# Assign Census variables vector to race_vars
race_vars <- c(White = "B03002_003", Black = "B03002_004", Native = "B03002_005",
               Asian = "B03002_006", HIPI = "B03002_007", Hispanic = "B03002_012")
```

```r
# Request a summary variable from the ACS
ca_race <- get_acs(geography = "county",
                   state = "CA",
                   variables = race_vars,
                   summary_var = "B03002_001")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```r
# Calculate a new percentage column and check the result
ca_race_pct <- ca_race %>%
  mutate(pct = 100 * (estimate / summary_est))

head(ca_race_pct)
```

```
## # A tibble: 6 x 8
##   GEOID NAME              variable estimate   moe summary_est summary_moe    pct
##   <chr> <chr>             <chr>       <dbl> <dbl>       <dbl>       <dbl>  <dbl>
## 1 06001 Alameda County, ~ White      466445  1170     1651949          NA 28.2
## 2 06001 Alameda County, ~ Black      159042  1736     1651949          NA  9.63
## 3 06001 Alameda County, ~ Native       4002   418     1651949          NA  0.242
## 4 06001 Alameda County, ~ Asian      528377  2269     1651949          NA 32.0
## 5 06001 Alameda County, ~ HIPI        11651   588     1651949          NA  0.705
## 6 06001 Alameda County, ~ Hispanic   385245    NA     1651949          NA 23.3
```

```r
# Group the dataset and filter the estimate
ca_largest <- ca_race %>%
  group_by(GEOID) %>%
  filter(estimate == max(estimate))

head(ca_largest)
```

```
## # A tibble: 6 x 7
## # Groups:   GEOID [6]
##   GEOID NAME                     variable estimate   moe summary_est summary_moe
##   <chr> <chr>                    <chr>       <dbl> <dbl>       <dbl>       <dbl>
## 1 06001 Alameda County, Califor~ Asian      528377  2269     1651949          NA
## 2 06003 Alpine County, Californ~ White         993   215        1695         234
## 3 06005 Amador County, Californ~ White       30234   341       41029          NA
## 4 06007 Butte County, California White      139527   767      209470          NA
## 5 06009 Calaveras County, Calif~ White       35599   318       45995          NA
## 6 06011 Colusa County, Californ~ Hispanic    13639    NA       21895          NA
```

```r
# Group the dataset and get a breakdown of the results
ca_largest %>%
  group_by(GEOID) %>%
  tally()
```

```
## # A tibble: 58 x 2
##    GEOID     n
##    <chr> <int>
```

```
##  1 06001     1
##  2 06003     1
##  3 06005     1
##  4 06007     1
##  5 06009     1
##  6 06011     1
##  7 06013     1
##  8 06015     1
##  9 06017     1
## 10 06019     1
## # i 48 more rows
```

```
# Use a tidy workflow to wrangle ACS data
wa_grouped <- wa_income %>%
  filter(variable != "B19001_001") %>%
  mutate(incgroup = case_when(
    variable < "B19001_008" ~ "below35k",
    variable < "B19001_013" ~ "35kto75k",
    TRUE ~ "above75k"
  )) %>%
  group_by(NAME, incgroup) %>%
  summarize(group_est = sum(estimate))
```

```
## `summarise()` has grouped output by 'NAME'. You can override using the
## `.groups` argument.
```

```
wa_grouped
```

```
## # A tibble: 117 x 3
## # Groups:   NAME [39]
##    NAME                       incgroup group_est
##    <chr>                      <chr>        <dbl>
##  1 Adams County, Washington   35kto75k      1834
##  2 Adams County, Washington   above75k      2690
##  3 Adams County, Washington   below35k      1777
##  4 Asotin County, Washington 35kto75k      2643
##  5 Asotin County, Washington above75k      4380
##  6 Asotin County, Washington below35k      2415
##  7 Benton County, Washington 35kto75k     19535
##  8 Benton County, Washington above75k     43699
##  9 Benton County, Washington below35k     13462
## 10 Chelan County, Washington 35kto75k      8787
## # i 107 more rows
```

```
# Map through ACS1 estimates to see how they change through the years
mi_cities <- map_df(2012:2016, function(x) {
  get_acs(geography = "place",
          variables = c(totalpop = "B01003_001"),
          state = "MI",
          survey = "acs1",
          year = x) %>%
    mutate(year = x)
})
```

```
## Getting data from the 2012 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
## Getting data from the 2013 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
## Getting data from the 2014 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
## Getting data from the 2015 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
## Getting data from the 2016 1-year ACS
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
```

```
mi_cities %>% arrange(NAME, year)
```

```
## # A tibble: 80 x 6
##    GEOID   NAME                     variable estimate   moe  year
##    <chr>   <chr>                    <chr>       <dbl> <dbl> <int>
##  1 2603000 Ann Arbor city, Michigan totalpop   116128    35  2012
##  2 2603000 Ann Arbor city, Michigan totalpop   117034    43  2013
##  3 2603000 Ann Arbor city, Michigan totalpop   117759    44  2014
##  4 2603000 Ann Arbor city, Michigan totalpop   117070    33  2015
##  5 2603000 Ann Arbor city, Michigan totalpop   120777    33  2016
##  6 2621000 Dearborn city, Michigan  totalpop    96470    28  2012
##  7 2621000 Dearborn city, Michigan  totalpop    95888    35  2013
##  8 2621000 Dearborn city, Michigan  totalpop    95546    48  2014
##  9 2621000 Dearborn city, Michigan  totalpop    95180    40  2015
## 10 2621000 Dearborn city, Michigan  totalpop    94430    52  2016
## # i 70 more rows
```

```
# Get data on elderly poverty by Census tract in Vermont
vt_eldpov <- get_acs( geography = "tract",
                      variables = c(eldpovm = "B17001_016",
                                    eldpovf = "B17001_030"),
                      state = "VT")
```

```
## Getting data from the 2019-2023 5-year ACS
```

```
vt_eldpov
```

```
## # A tibble: 386 x 5
##    GEOID       NAME                                        variable estimate   moe
##    <chr>       <chr>                                       <chr>       <dbl> <dbl>
##  1 50001960100 Census Tract 9601; Addison County; Vermo~ eldpovm         6     9
##  2 50001960100 Census Tract 9601; Addison County; Vermo~ eldpovf         7     7
##  3 50001960200 Census Tract 9602; Addison County; Vermo~ eldpovm         0    10
##  4 50001960200 Census Tract 9602; Addison County; Vermo~ eldpovf         0    10
##  5 50001960300 Census Tract 9603; Addison County; Vermo~ eldpovm         0    10
##  6 50001960300 Census Tract 9603; Addison County; Vermo~ eldpovf         9    12
##  7 50001960400 Census Tract 9604; Addison County; Vermo~ eldpovm        10    11
##  8 50001960400 Census Tract 9604; Addison County; Vermo~ eldpovf         6     4
##  9 50001960500 Census Tract 9605; Addison County; Vermo~ eldpovm        10    14
## 10 50001960500 Census Tract 9605; Addison County; Vermo~ eldpovf        19    23
## # i 376 more rows
```

```r
# Identify rows with greater margins of error than their estimates
moe_check <- filter(vt_eldpov, moe > estimate)

# Check proportion of rows where the margin of error exceeds the estimate
nrow(moe_check) / nrow(vt_eldpov)
```

```
## [1] 0.7668394
```

```r
# Group the dataset and calculate a derived margin of error
vt_eldpov2 <- vt_eldpov %>%
  group_by(GEOID) %>%
  summarize(
    estmf = sum(estimate),
    moemf = moe_sum( moe = moe, estimate = estimate)
  )

# Filter rows where newly-derived margin of error exceeds newly-derived estimate
moe_check2 <- filter(vt_eldpov2,  moemf > estmf)

# Check proportion of rows where margin of error exceeds estimate
nrow(moe_check2) / nrow(vt_eldpov2)
```

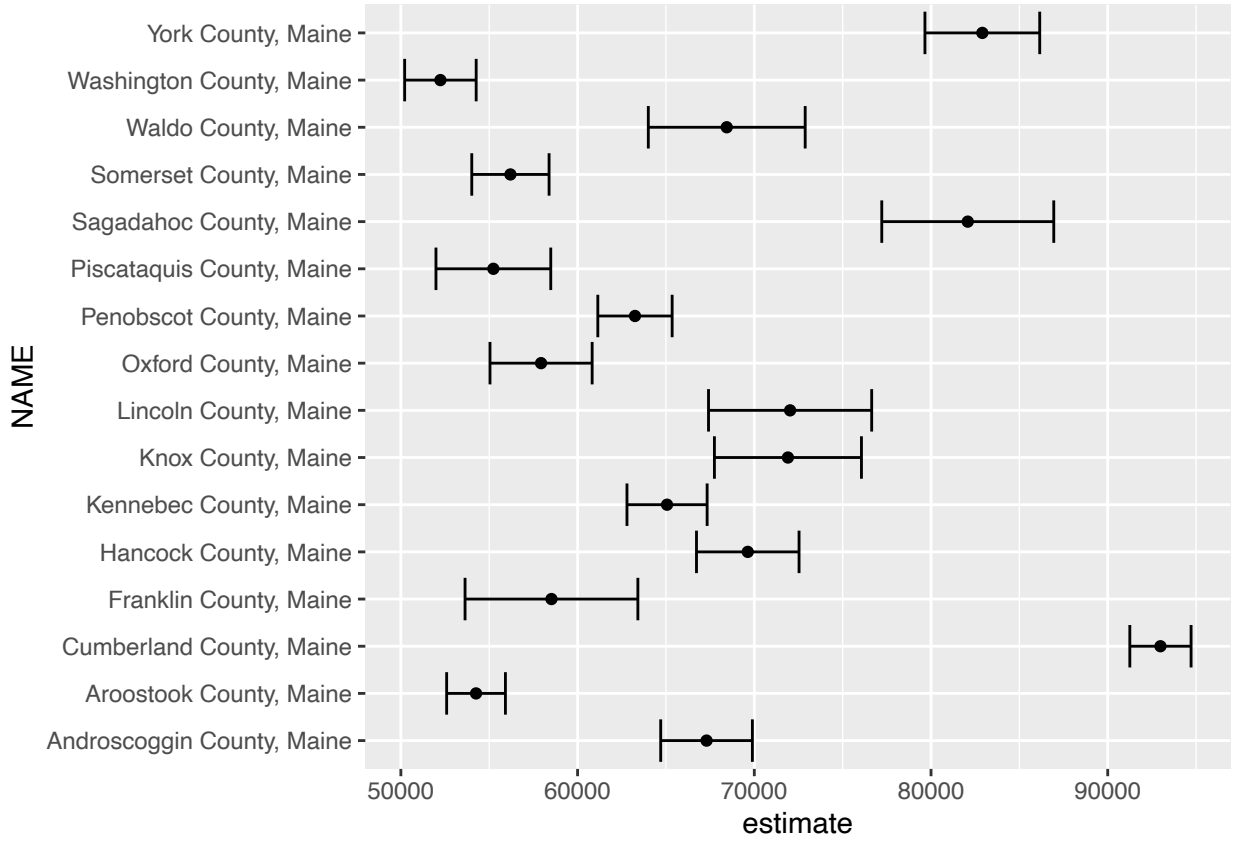
```
## [1] 0.5854922
```

```r
# Request median household income data
maine_inc <- get_acs(geography = "county",
                     variables = c(hhincome = "B19013_001"),
                     state = "ME")
```

```
## Getting data from the 2019-2023 5-year ACS
```

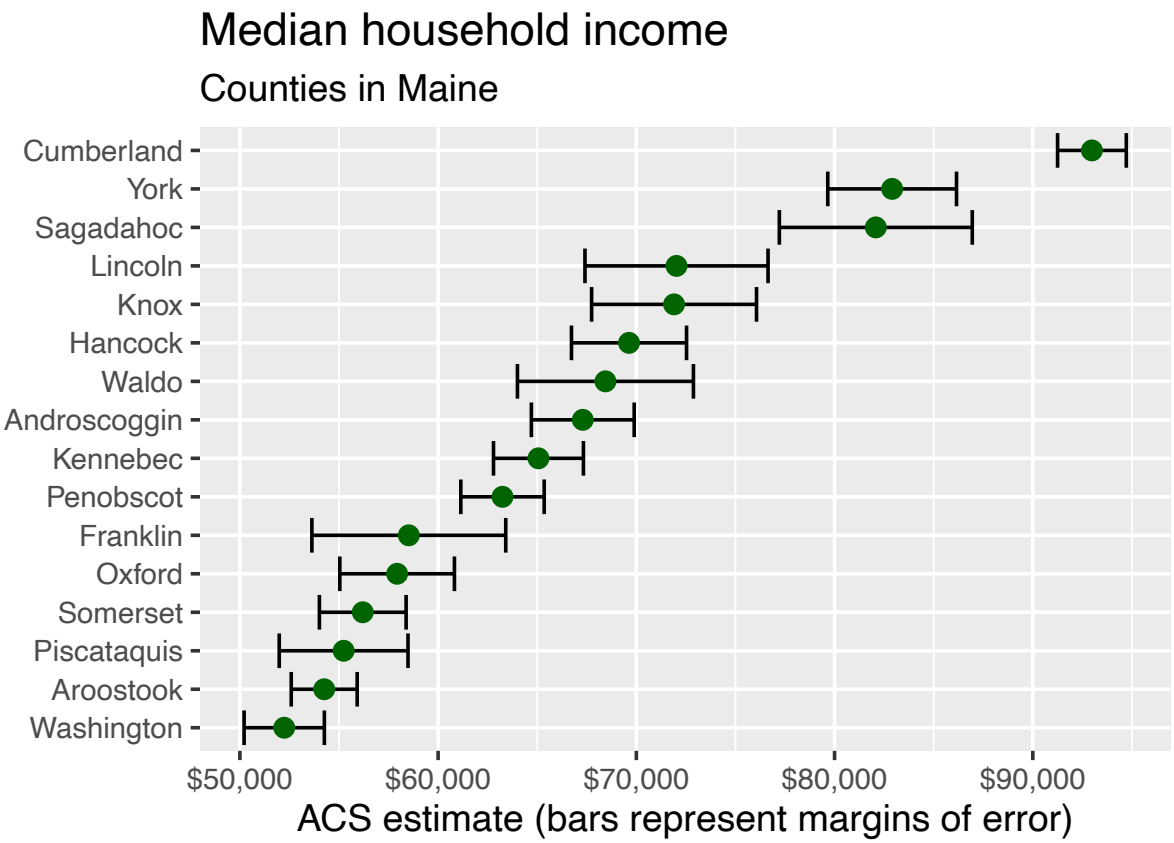
```r
# Generate horizontal error bars with dots
ggplot(maine_inc, aes(x = estimate, y = NAME)) +
  geom_errorbarh(aes(xmin = estimate - moe, xmax = estimate + moe)) +
  geom_point()
```

```
## Warning: `geom_errorbarh()` was deprecated in ggplot2 4.0.0.
## i Please use the `orientation` argument of `geom_errorbar()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# Remove unnecessary content from the county's name
maine_inc2 <- maine_inc %>%
  mutate(NAME = str_replace(NAME, " County, Maine", ""))

# Build a margin of error plot incorporating your modifications
ggplot(maine_inc2, aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_errorbarh(aes(xmin = estimate - moe, xmax = estimate + moe)) +
  geom_point(size = 3, color = "darkgreen") +
  theme_grey(base_size = 14) +
  labs(title = "Median household income",
       subtitle = "Counties in Maine",
       x = "ACS estimate (bars represent margins of error)",
       y = "") +
  scale_x_continuous(labels = scales::dollar)
```



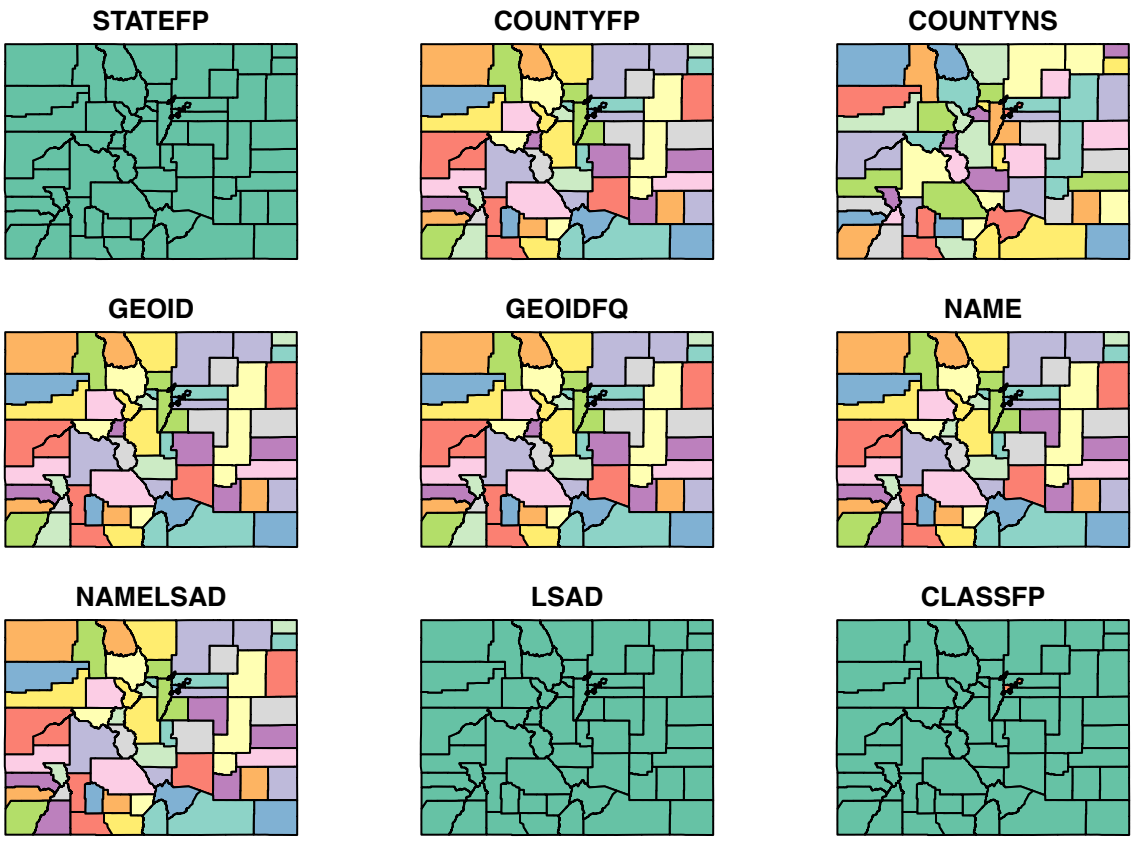

```
# Get a counties dataset for Colorado and plot it
co_counties <- counties(state = "CO")
```

```
## Retrieving data for the year 2024
```

```
##   |                                                                              |
```

```
plot(co_counties)
```

```
## Warning: plotting the first 9 out of 18 attributes; use max.plot = 18 to plot
## all
```



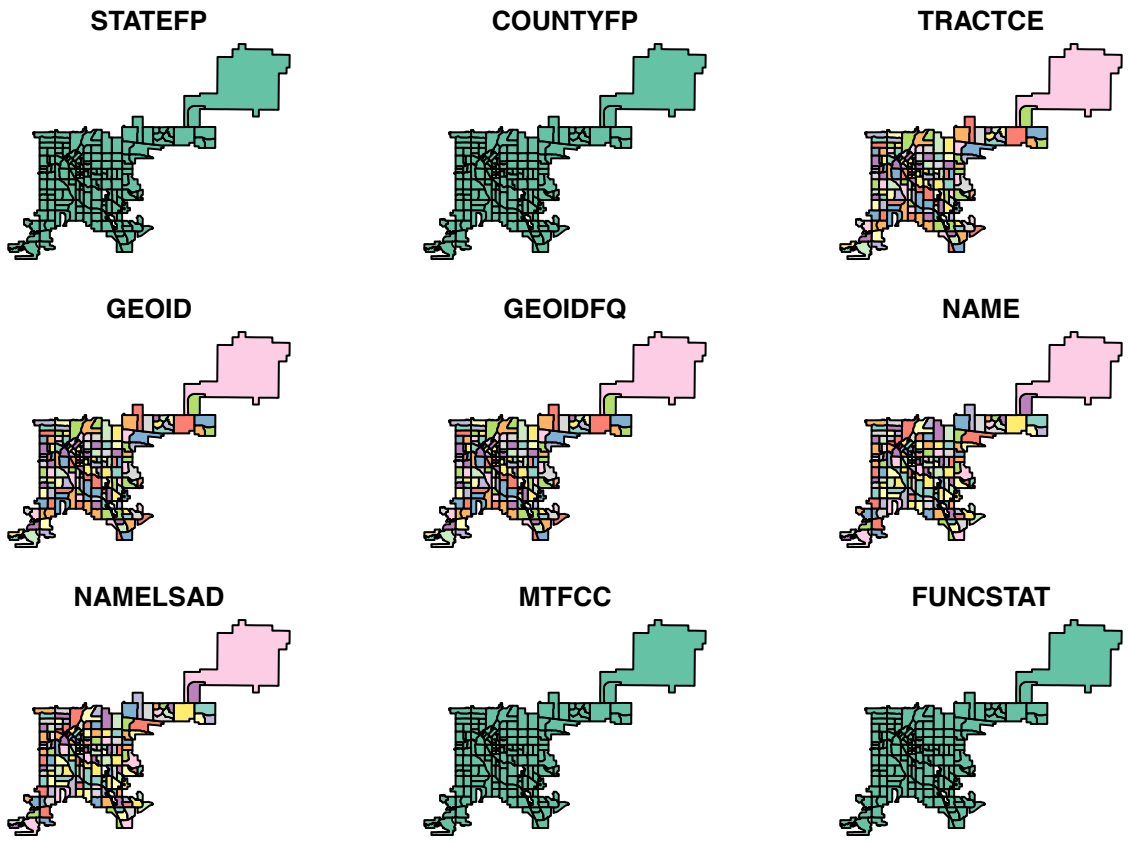

```
# Get a Census tracts dataset for Denver County, Colorado and plot it
denver_tracts <- tracts(state = "CO", county = "Denver")
```

```
## Retrieving data for the year 2024
```

```
##   |                                                                              |
```

```
plot(denver_tracts)
```

```
## Warning: plotting the first 9 out of 13 attributes; use max.plot = 13 to plot
## all
```

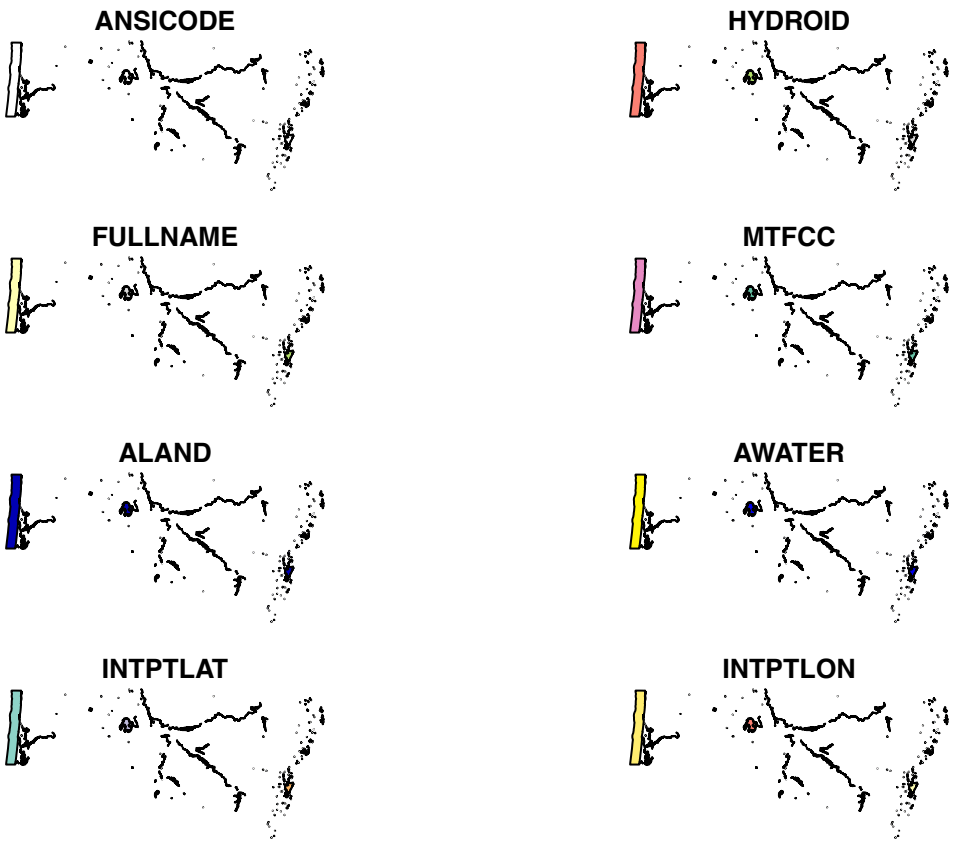



```
# Plot area water features for Lane County, Oregon
lane_water <- area_water(state = "OR", county = "Lane")
```

```
## Retrieving data for the year 2024
```

```
##   |                                                                              |
```

```
plot(lane_water)
```

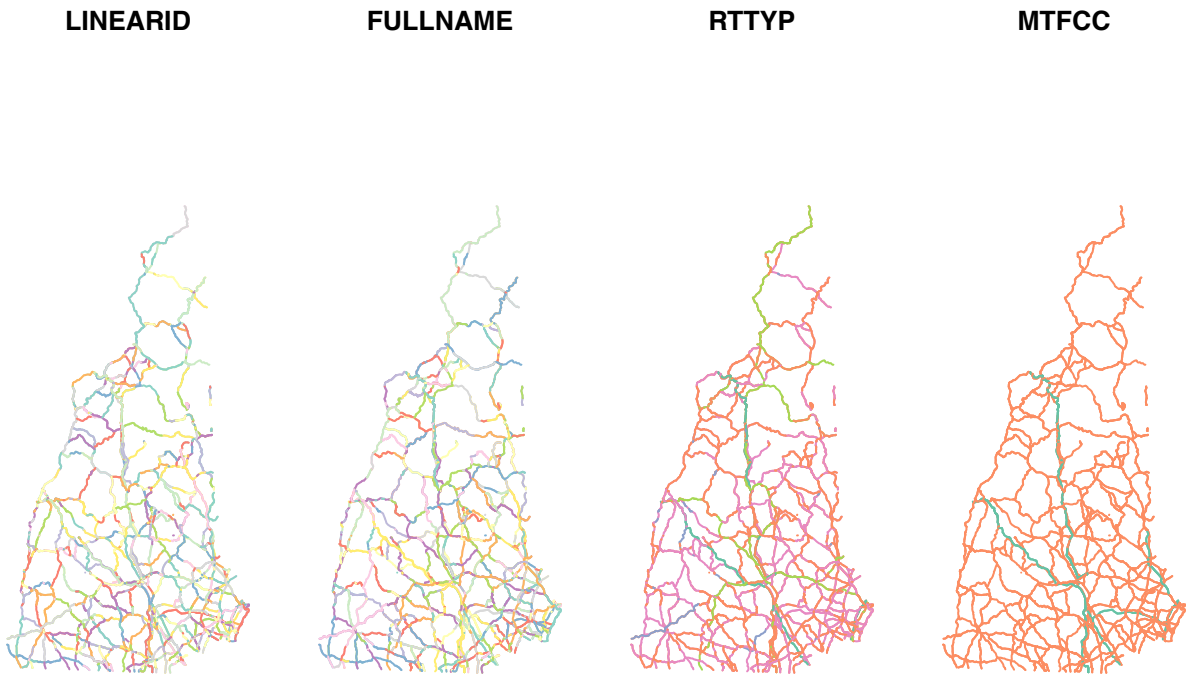



```
# Plot primary & secondary roads for the state of New Hampshire
nh_roads <- primary_secondary_roads(state = "NH")
```

```
## Retrieving data for the year 2024
```

```
##   |                                                                              |
```

```
plot(nh_roads)
```




```r
# Get a counties dataset for Michigan
mi_tiger <- counties("MI")
```

```
## Retrieving data for the year 2024
```

```r
# Get the equivalent cartographic boundary shapefile
mi_cb <- counties("MI", cb = TRUE)
```

```
## Retrieving data for the year 2024
```

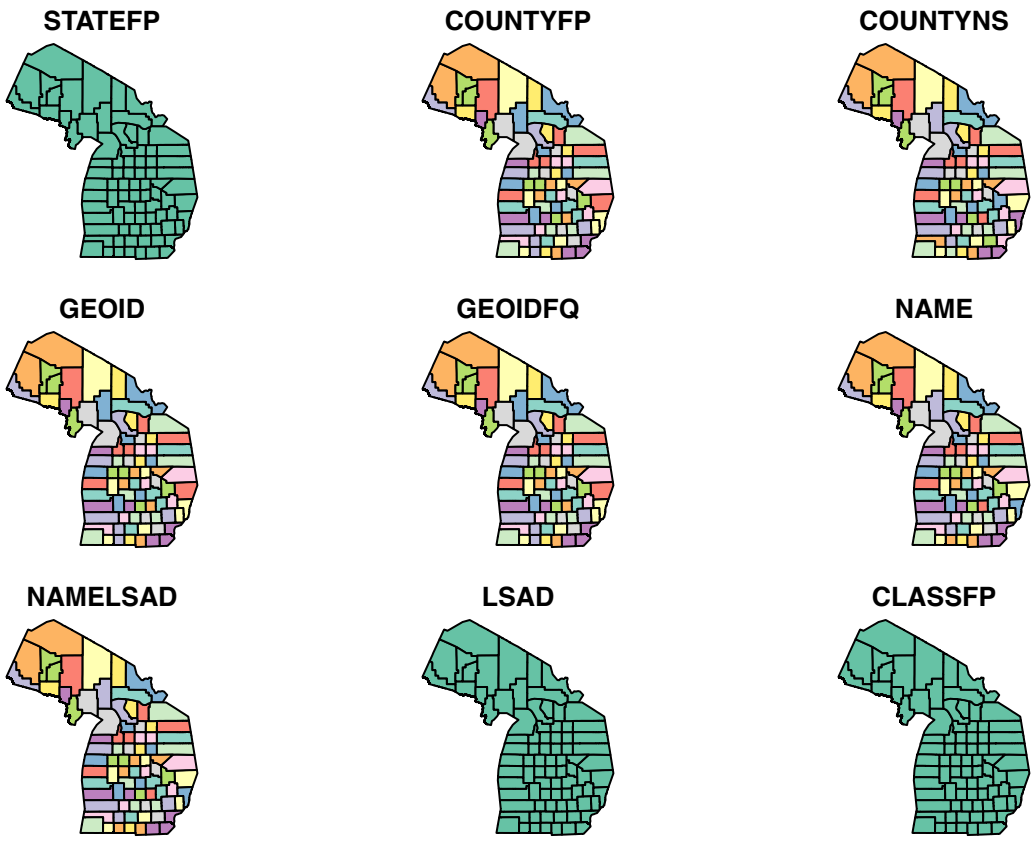
```
##   |                                                                              |
```

```r
# Overlay the two on a plot to make a comparison
plot(mi_tiger)
```

```
## Warning: plotting the first 9 out of 18 attributes; use max.plot = 18 to plot
## all
```

```r
plot(mi_cb, add = TRUE, border = "red")
```

```
## Warning in plot.sf(mi_cb, add = TRUE, border = "red"): ignoring all but the
## first attribute
```




```
#Get data from tigris as simple features
options(tigris_class = "sf")

# Get countries from Colorado and view the first few rows
colorado_sf <- counties("CO")
```

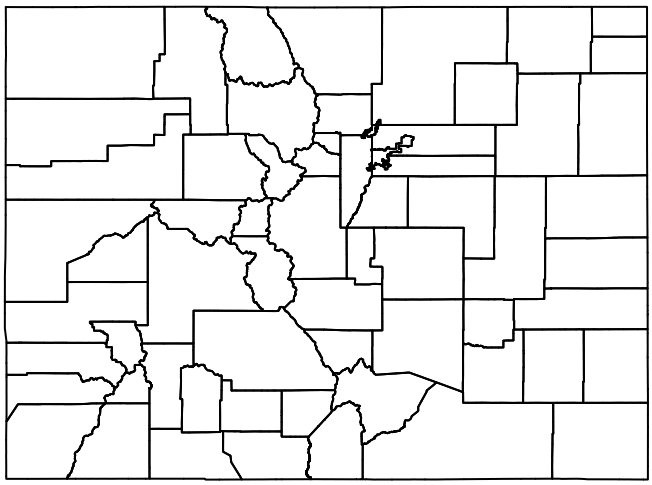
```
## Retrieving data for the year 2024
```

```
head(colorado_sf)
```

```
## Simple feature collection with 6 features and 18 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -108.3811 ymin: 36.99961 xmax: -102.0448 ymax: 41.0026
## Geodetic CRS:  NAD83
##     STATEFP COUNTYFP COUNTYNS GEOID        GEOIDFQ     NAME        NAMELSAD
## 23       08      109 00198170 08109 0500000US08109 Saguache Saguache County
## 106      08      115 00198173 08115 0500000US08115 Sedgwick Sedgwick County
## 123      08      017 00198124 08017 0500000US08017 Cheyenne Cheyenne County
## 162      08      027 00198129 08027 0500000US08027   Custer   Custer County
## 199      08      067 00198148 08067 0500000US08067 La Plata La Plata County
## 227      08      111 00198171 08111 0500000US08111 San Juan San Juan County
##     LSAD CLASSFP MTFCC CSAFP CBSAFP METDIVFP FUNCSTAT      ALAND   AWATER
## 23    06      H1 G4020  <NA>   <NA>     <NA>        A 8206547700  4454510
## 106   06      H1 G4020  <NA>   <NA>     <NA>        A 1419419024  3530746
```

```
## 123   06      H1 G4020  <NA>   <NA>     <NA>        A 4605713958  8166129
## 162   06      H1 G4020  <NA>   <NA>     <NA>        A 1913031985  3364150
## 199   06      H1 G4020  <NA>  20420     <NA>        A 4376255274 25642579
## 227   06      H1 G4020  <NA>   <NA>     <NA>        A 1003660671  2035929
##        INTPTLAT     INTPTLON                       geometry
## 23  +38.0316514 -106.2346662 MULTIPOLYGON (((-107.0019 3...
## 106 +40.8715679 -102.3553579 MULTIPOLYGON (((-102.6535 4...
## 123 +38.8356456 -102.6017914 MULTIPOLYGON (((-102.547 38...
## 162 +38.1019955 -105.3735123 MULTIPOLYGON (((-105.7969 3...
## 199 +37.2873673 -107.8397178 MULTIPOLYGON (((-107.7124 3...
## 227 +37.7810492 -107.6702567 MULTIPOLYGON (((-107.9751 3...
```

```r
# Plot its geometry column
plot(colorado_sf$geometry)
```



```r
# Set the cache directory
tigris_cache_dir("~/Desktop/Projects/SQL projects/github")
```

```
## Your new tigris cache directory is ~/Desktop/Projects/SQL projects/github.
## To use now, restart R or run `readRenviron('~/.Renviron')`
```

```r
# Set the tigris_use_cache option
options(tigris_use_cache = TRUE)
```

```r
# Check to see that you've modified the option correctly
getOption("tigris_use_cache")
```

```
## [1] TRUE
```

```r
# Get a historic Census tract shapefile from 1990 for Williamson County, Texas
williamson90 <- tracts(state = "TX", county = "Williamson",
                       cb = TRUE, year = 1990)

# Compare with a current dataset for 2016
williamson16 <- tracts(state = "TX", county = "Williamson",
                       cb = TRUE, year = 2016)

# Plot the geometry to compare the results
par(mfrow = c(1, 2))
plot(williamson90$geometry)
plot(williamson16$geometry)
```

```r
# Get Census tract boundaries for Oregon and Washington
or_tracts <- tracts("OR", cb = TRUE)
```

```
## Retrieving data for the year 2024
```

```r
wa_tracts <- tracts("WA", cb = TRUE)
```

```
## Retrieving data for the year 2024
```

```r
# Check the tigris attributes of each object
attr(or_tracts, "tigris")
```

```
## [1] "tract"
```

```r
attr(wa_tracts, "tigris")
```

```
## [1] "tract"
```

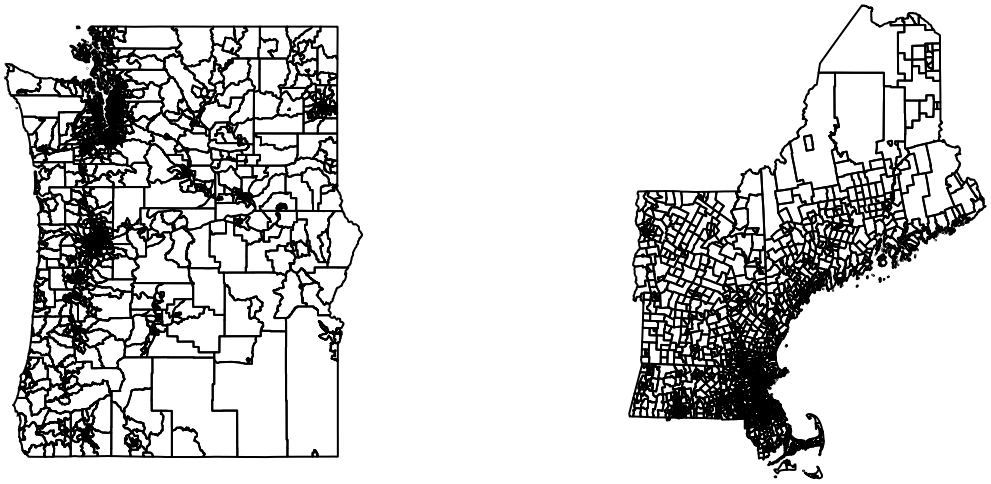
```r
# Combine the datasets then plot the result
or_wa_tracts <- rbind_tigris(or_tracts, wa_tracts)
plot(or_wa_tracts$geometry)

# Generate a vector of state codes and assign to new_england
new_england <- c("ME", "NH", "VT", "MA")

# Iterate through the states and request tract data for state
ne_tracts <- map(new_england, function(x) {
  tracts(state = x, cb = TRUE)
}) %>%
  rbind_tigris()
```

```
## Retrieving data for the year 2024
## Retrieving data for the year 2024
## Retrieving data for the year 2024
## Retrieving data for the year 2024
```

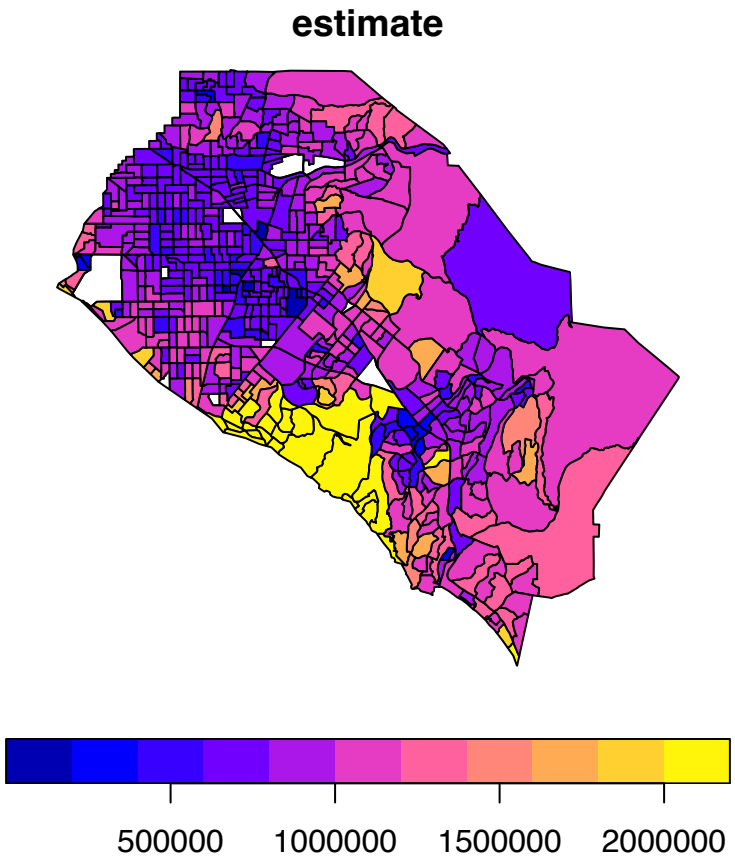
```r
plot(ne_tracts$geometry)
```



```
# Get dataset with geometry set to TRUE
orange_value <- get_acs(geography = "tract", state = "CA",
                        county = "Orange",
                        variables = "B25077_001",
                        geometry = TRUE)
```

```
## Getting data from the 2019-2023 5-year ACS
```

```
# Plot the estimate to view a map of the data
plot(orange_value["estimate"])
```




```
# Get an income dataset for Idaho by school district
idaho_income <- get_acs(geography ="school district (unified)",
                        variables = "B19013_001",
                        state = "ID")
```

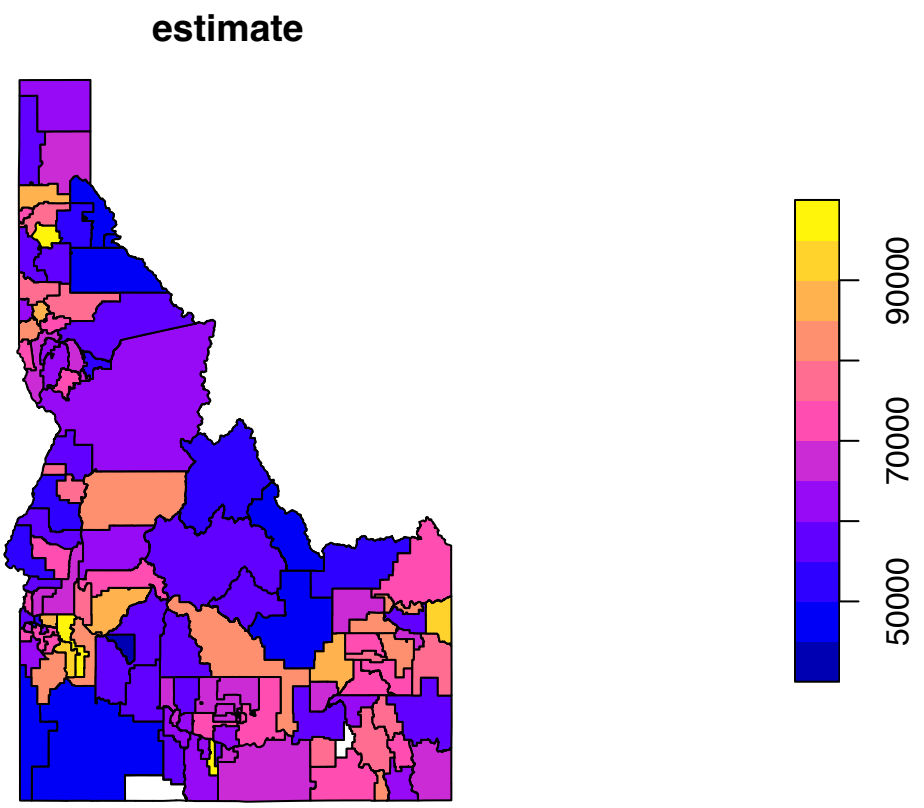
```
## Getting data from the 2019-2023 5-year ACS
```

```
# Get a school district dataset for Idaho
idaho_school <- school_districts(state = "ID", type = "unified", class = "sf")
```

```
## Retrieving data for the year 2024
```

```
# Join the income dataset to the boundaries dataset
id_school_joined <- left_join(idaho_school, idaho_income, by = "GEOID")

plot(id_school_joined["estimate"])
```



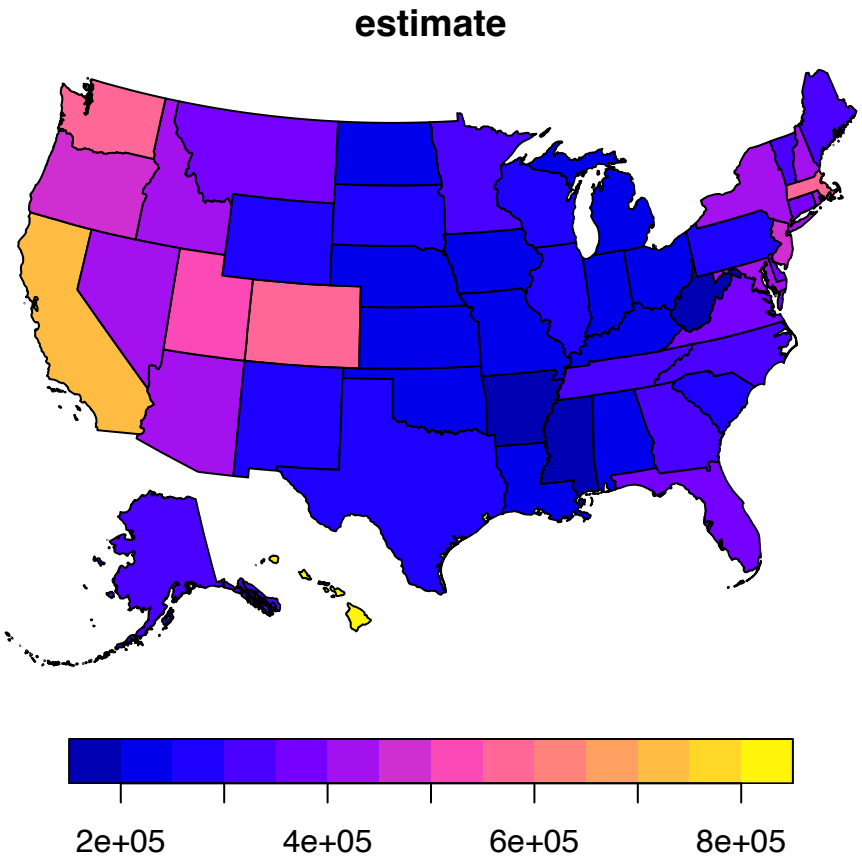

```
# Get a dataset of median home values from the 1-year ACS
state_value <- get_acs(geography = "state",
                       variables = "B25077_001",
                       survey = "acs1",
                       geometry = TRUE,
                       shift_geo = TRUE)
```

```
## Getting data from the 2023 1-year ACS
```

```
## Warning: The ‘shift_geo‘ argument is deprecated and will be removed in a future
## release. We recommend using ‘tigris::shift_geometry()‘ instead.
```

```
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
## Using feature geometry obtained from the albersusa package
## Please note: Alaska and Hawaii are being shifted and are not to scale.
## old-style crs object detected; please recreate object with a recent sf::st_crs()
```

```
# Plot the dataset to view the shifted geometry
plot(state_value["estimate"])
```



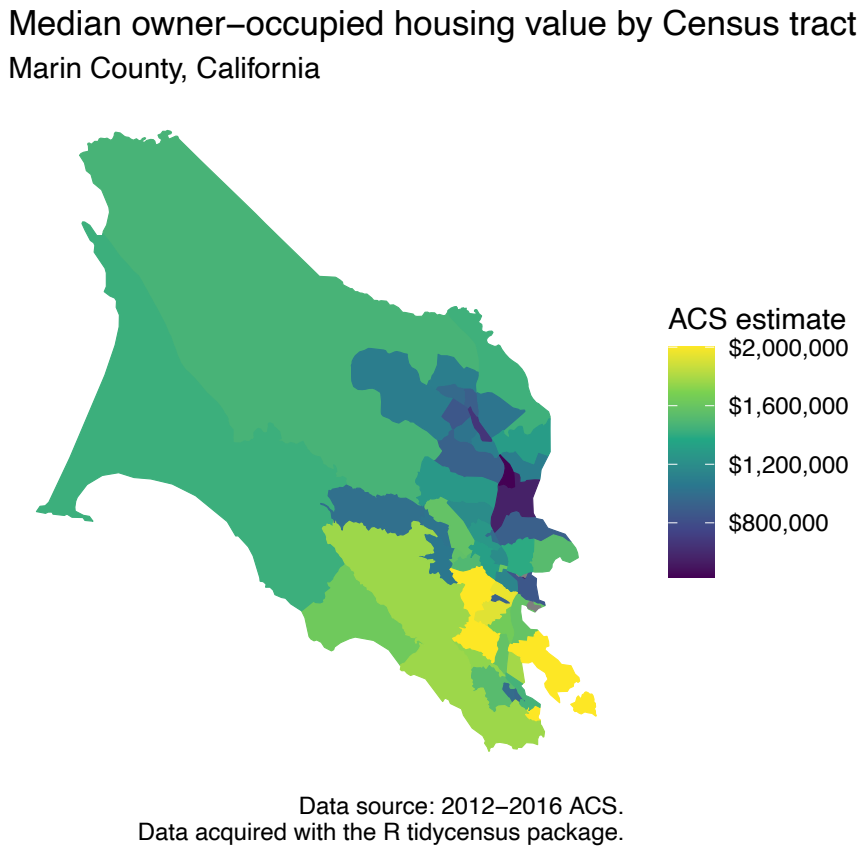

```
marin_value <- get_acs(geography = "tract",
                       variables = "B25077_001",
                       state = "CA",
                       county = "Marin",
                       geometry = TRUE,
                       year = 2022)
```

```
## Getting data from the 2018-2022 5-year ACS
```

```
# Set the color guide to FALSE and add a subtitle and caption to your map
ggplot(marin_value, aes(fill = estimate, color = estimate)) +
  geom_sf() +
  scale_fill_viridis_c(labels = scales::dollar) +
  scale_color_viridis_c(guide = FALSE) +
  theme_minimal() +
  coord_sf(crs = 26911, datum = NA) +
  labs(title = "Median owner-occupied housing value by Census tract",
       subtitle = "Marin County, California",
       caption = "Data source: 2012-2016 ACS.\nData acquired with the R tidycensus package.",
       fill = "ACS estimate")
```

```
## Warning: The ‘guide‘ argument in ‘scale_*()‘ cannot be ‘FALSE‘. This was deprecated in
## ggplot2 3.3.4.
## i Please use "none" instead.
## This warning is displayed once every 8 hours.
```

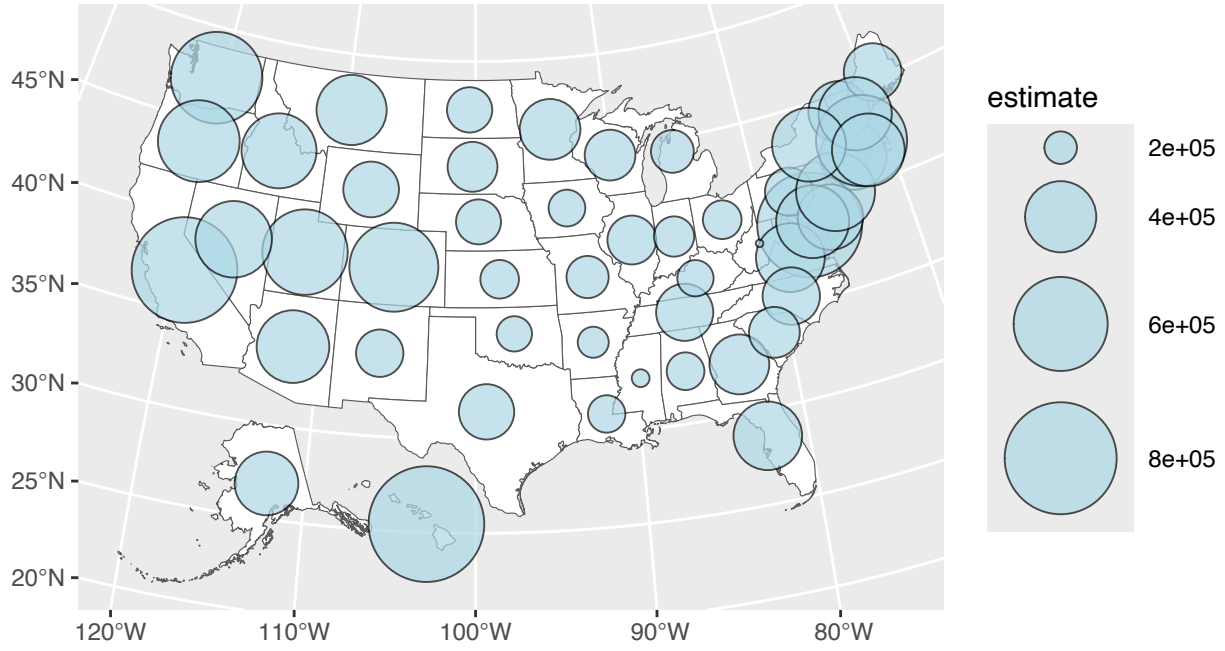
```
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




```
# Generate point centers
centers <- st_centroid(state_value)
```

```
## Warning: st_centroid assumes attributes are constant over geometries
```

```
# Set size parameter and the size range
ggplot() +
  geom_sf(data = state_value, fill = "white") +
  geom_sf(data = centers, aes(size = estimate), shape = 21,
          fill = "lightblue", alpha = 0.7, show.legend = "point") +
  scale_size_continuous(range = c(1, 20))
```




```
######
# Pull race data for DC Census tracts (with geometry)
dc_race_raw <- get_acs(
  geography = "tract",
  state = "DC",
  variables = race_vars,
  year = 2022,
  geometry = TRUE
)
```

```
## Getting data from the 2018-2022 5-year ACS
```

```
dc_race <- dc_race_raw %>%
  select(GEOID, NAME, variable, estimate, geometry) %>%
  rename(value = estimate) %>%
  group_by(GEOID) %>%
  mutate(
    # Compute total population per tract by summing race counts
    summary_value = sum(value[variable != "Total"], na.rm = TRUE),
    # Calculate percentage share per race
    percent = (value / summary_value) * 100
  ) %>%
  # Remove the total row; keep race-level data only
  filter(variable != "Total") %>%
  ungroup()
```

```
######
dc_dots <- map(c("White", "Black", "Hispanic", "Asian"), function(group) {
  dc_race %>%
    filter(variable == group) %>%
    st_sample(., size = .$value / 100) %>%
    st_sf() %>%
    mutate(group = group)
}) %>%
  reduce(rbind) %>%
  group_by(group) %>%
  summarize()
```

```
## Warning in st_sample.sfc(st_geometry(x), size, ...): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer

## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer

## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
```

```
## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
```

```
## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
```

```
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
```

```
## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x[i], size[i], type = type, exact = exact, ...): size
## is not an integer
```

```
## Warning in st_sample.sfc(x = x, size = size, ..., type = type, exact = FALSE):
## size is not an integer
```

```
## Warning in st_sample.sfc(x, size = diff, ..., type, exact = FALSE, by_polygon =
## by_polygon): size is not an integer
```

```
# Filter the DC roads object for major roads only
dc_roads <- roads("DC", "District of Columbia") %>%
  filter(RTTYP %in% c("I", "S", "U"))

# Get an area water dataset for DC
dc_water <- area_water("DC", "District of Columbia")
```

```
## Retrieving data for the year 2024
```

```r
# Get the boundary of DC
dc_boundary <- counties("DC", cb = TRUE)
```

```
## Retrieving data for the year 2024
```

```r
# Plot your datasets and give your map an informative caption
ggplot() +
  geom_sf(data = dc_boundary, color = NA, fill = "white") +
  geom_sf(data = dc_dots, aes(color = group, fill = group), size = 0.1) +
  geom_sf(data = dc_boundary, color = "lightblue", fill = "lightblue") +
  geom_sf(data = dc_dots, color = "grey") +
  coord_sf(crs = 26918, datum = NA) +
  scale_color_brewer(palette = "Set1", guide = FALSE) +
  scale_fill_brewer(palette = "Set1") +
  labs(title = "The racial geography of Washington, DC",
       subtitle = "2010 decennial U.S. Census",
       fill = "",
       caption = "1 dot = approximately 100 people.\nData acquired with the R tidycensus and tigris pack
```