An introduction to tidycensus

Ram Mandava

2023-02-17

1. What worked well for you and what didn't work?

Analyzed the US Census Data with help of existing R Studio code chunks and produce in terms of plots, tables, and caching data. These complex tasks can be segregated into sections to produce the results and converge in a publication (along with supplementary materials). Final "enriched document", can back-trace and reproduce the entire analysis made or just part of it.

If we have a good understanding of R programming, the maintainability and **reproducibility** of outcomes will be much easier to handle.

But some of the R code chunks didn't work and gave the below errors and were unable to fix the issues.

Examples:

1. The API message returned is the error: unknown variable 'P01001'.
2. Error in Use Method("gather"): no applicable method for 'gather' applied to an object of class "character".

2. What do you see as the value of using RMarkdown?

R Markdown is a subset of literate programming, which aims to make it simple to write reproducible web-based reports by combining text and source code into a single document. Used to create documents in many different formats, including HTML, PDF, and MS Word.

Ability to break down into sections that give better readability. Both papers and supplementary materials can be enriched by documents that keep track of all code, data, and results produced during the analysis.

3. Based on lab 2 so far, what things would you like to do?

Familiarized with the United States Census and the US Census Bureau, and how Census data can be accessed and used by analysts.

Importance of R package for working with US Census Bureau data in a tidy format.

Understanding the basic data requests with the package and various options in the package.

R programming techniques to streamline the data wrangling process for spatial Census data analysis and explore US Census data with visualization. Workflows and best practices for preparing data for visualization and building charts for presenting Census data analyses.

Getting started with tidycensus code chunks with results

```
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.2.2
```

```
#census_api_key("1840a214a15b3834c5308103bdaf17009f4db1b4", install = TRUE)
```

```
total_population_10 <- get_decennial(
  geography = "state",
  variables = "P001001",
  year = 2010
)
```

```
## Getting data from the 2010 decennial Census
```

```
## Using Census Summary File 1
```

```
print(total_population_10)
```

```
## # A tibble: 52 × 4
##    GEOID NAME        variable    value
##    <chr> <chr>       <chr>       <dbl>
##  1 01    Alabama     P001001   4779736
##  2 02    Alaska      P001001    710231
##  3 04    Arizona     P001001   6392017
##  4 05    Arkansas    P001001   2915918
##  5 06    California  P001001  37253956
##  6 22    Louisiana   P001001   4533372
##  7 21    Kentucky    P001001   4339367
##  8 08    Colorado    P001001   5029196
##  9 09    Connecticut P001001   3574097
## 10 10    Delaware    P001001    897934
## # … with 42 more rows
```

```
aian_2020 <- get_decennial(
  geography = "state",
  variables = "P1_005N",
  year = 2020,
  sumfile = "pl"
)
```

```
## Getting data from the 2020 decennial Census
```

```
## Using the PL 94-171 Redistricting Data summary file
```

```
## Note: 2020 decennial Census data use differential privacy, a technique that
## introduces errors into data to preserve respondent confidentiality.
## ℹ Small counts should be interpreted with caution.
## ℹ See https://www.census.gov/library/fact-sheets/2021/protecting-the-confidentiality-of-the-2
020-census-redistricting-data.html for additional guidance.
## This message is displayed once per session.
```

```
print(aian_2020)
```

```
## # A tibble: 52 × 4
##    GEOID NAME                 variable  value
##    <chr> <chr>                <chr>     <dbl>
##  1 42    Pennsylvania         P1_005N   31052
##  2 06    California           P1_005N  631016
##  3 54    West Virginia        P1_005N    3706
##  4 49    Utah                 P1_005N   41644
##  5 36    New York             P1_005N  149690
##  6 11    District of Columbia P1_005N    3193
##  7 02    Alaska               P1_005N  111575
##  8 12    Florida              P1_005N   94795
##  9 45    South Carolina       P1_005N   24303
## 10 38    North Dakota         P1_005N   38914
## # … with 42 more rows
```

```
born_in_mexico <- get_acs(
  geography = "state",
  variables = "B05006_150",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
print(born_in_mexico)
```

```
## # A tibble: 52 × 5
##    GEOID NAME                 variable   estimate   moe
##    <chr> <chr>                <chr>         <dbl> <dbl>
##  1 01    Alabama              B05006_150    46927  1846
##  2 02    Alaska               B05006_150     4181   709
##  3 04    Arizona              B05006_150   510639  8028
##  4 05    Arkansas             B05006_150    60236  2182
##  5 06    California           B05006_150  3962910 25353
##  6 08    Colorado             B05006_150   215778  4888
##  7 09    Connecticut          B05006_150    28086  2144
##  8 10    Delaware             B05006_150    14616  1065
##  9 11    District of Columbia B05006_150     4026   761
## 10 12    Florida              B05006_150   257933  6418
## # … with 42 more rows
```

```
born_in_mexico_1yr <- get_acs(
  geography = "state",
  variables = "B05006_150",
  survey = "acs1",
  year = 2019
)
```

```
## Getting data from the 2019 1-year ACS
```

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

```
print(born_in_mexico_1yr)
```

```
## # A tibble: 52 × 5
##    GEOID NAME                 variable   estimate   moe
##    <chr> <chr>                <chr>         <dbl> <dbl>
##  1 01    Alabama              B05006_150       NA    NA
##  2 02    Alaska               B05006_150       NA    NA
##  3 04    Arizona              B05006_150   516618 15863
##  4 05    Arkansas             B05006_150       NA    NA
##  5 06    California           B05006_150  3951224 40506
##  6 08    Colorado             B05006_150   209408 12214
##  7 09    Connecticut          B05006_150    26371  4816
##  8 10    Delaware             B05006_150       NA    NA
##  9 11    District of Columbia B05006_150       NA    NA
## 10 12    Florida              B05006_150   261614 17571
## # … with 42 more rows
```

```
age_table <- get_acs(
  geography = "state",
  table = "B01001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## Loading ACS5 variables for 2020 from table B01001. To cache this dataset for faster access to ACS tables in the future, run this function with `cache_table = TRUE`. You only need to do this once per ACS dataset.
```

```
print(age_table)
```

```
## # A tibble: 2,548 × 5
##    GEOID NAME    variable   estimate   moe
##    <chr> <chr>   <chr>         <dbl> <dbl>
##  1 01    Alabama B01001_001  4893186    NA
##  2 01    Alabama B01001_002  2365734  1090
##  3 01    Alabama B01001_003   149579   672
##  4 01    Alabama B01001_004   150937  2202
##  5 01    Alabama B01001_005   160287  2159
##  6 01    Alabama B01001_006    96832   565
##  7 01    Alabama B01001_007    65459   961
##  8 01    Alabama B01001_008    36705  1467
##  9 01    Alabama B01001_009    33089  1547
## 10 01    Alabama B01001_010    93871  2045
## # … with 2,538 more rows
```

```
cbsa_population <- get_acs(
  geography = "cbsa",
  variables = "B01003_001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## Getting data from the 2016-2020 5-year ACS
print(cbsa_population)
```

```
## # A tibble: 939 × 5
##    GEOID NAME                                variable   estimate   moe
##    <chr> <chr>                               <chr>         <dbl> <dbl>
##  1 10100 Aberdeen, SD Micro Area             B01003_001    42864    NA
##  2 10140 Aberdeen, WA Micro Area             B01003_001    73769    NA
##  3 10180 Abilene, TX Metro Area              B01003_001   171354    NA
##  4 10220 Ada, OK Micro Area                  B01003_001    38385    NA
##  5 10300 Adrian, MI Micro Area               B01003_001    98310    NA
##  6 10380 Aguadilla-Isabela, PR Metro Area    B01003_001   295172    NA
##  7 10420 Akron, OH Metro Area                B01003_001   703286    NA
##  8 10460 Alamogordo, NM Micro Area           B01003_001    66804    NA
##  9 10500 Albany, GA Metro Area               B01003_001   147431    NA
## 10 10540 Albany-Lebanon, OR Metro Area       B01003_001   127216    NA
## # … with 929 more rows
```

```
wi_income <- get_acs(
  geography = "county",
  variables = "B19013_001",
  state = "WI",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
print(wi_income)
```

```
## # A tibble: 72 × 5
##    GEOID NAME                         variable   estimate   moe
##    <chr> <chr>                        <chr>         <dbl> <dbl>
##  1 55001 Adams County, Wisconsin      B19013_001    48906  2387
##  2 55003 Ashland County, Wisconsin    B19013_001    47869  3190
##  3 55005 Barron County, Wisconsin     B19013_001    52346  2092
##  4 55007 Bayfield County, Wisconsin   B19013_001    57257  2496
##  5 55009 Brown County, Wisconsin      B19013_001    64728  1419
##  6 55011 Buffalo County, Wisconsin    B19013_001    58364  1871
##  7 55013 Burnett County, Wisconsin    B19013_001    53555  2513
##  8 55015 Calumet County, Wisconsin    B19013_001    76065  2314
##  9 55017 Chippewa County, Wisconsin   B19013_001    61215  2064
## 10 55019 Clark County, Wisconsin      B19013_001    54463  1089
## # … with 62 more rows
```

```
dane_income <- get_acs(
  geography = "tract",
  variables = "B19013_001",
  state = "WI",
  county = "Dane",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
print(dane_income)
```

```
## # A tibble: 125 × 5
##    GEOID       NAME                                       variable  estim…¹   moe
##    <chr>       <chr>                                      <chr>       <dbl> <dbl>
##  1 55025000100 Census Tract 1, Dane County, Wisconsin    B19013_0…   74054 15662
##  2 55025000201 Census Tract 2.01, Dane County, Wisconsin B19013_0…   92460 27067
##  3 55025000202 Census Tract 2.02, Dane County, Wisconsin B19013_0…   88092  5189
##  4 55025000204 Census Tract 2.04, Dane County, Wisconsin B19013_0…   82717 12175
##  5 55025000205 Census Tract 2.05, Dane County, Wisconsin B19013_0…  100000 17506
##  6 55025000301 Census Tract 3.01, Dane County, Wisconsin B19013_0…   37016 11524
##  7 55025000302 Census Tract 3.02, Dane County, Wisconsin B19013_0…  117321 28723
##  8 55025000401 Census Tract 4.01, Dane County, Wisconsin B19013_0…  100434 12108
##  9 55025000402 Census Tract 4.02, Dane County, Wisconsin B19013_0…  105850 12205
## 10 55025000406 Census Tract 4.06, Dane County, Wisconsin B19013_0…   74009  2811
## # … with 115 more rows, and abbreviated variable name ¹estimate
```

```
nrow(wi_income)
```

```
## [1] 72
```

```
wi_income_1yr <- get_acs(
  geography = "county",
  variables = "B19013_001",
  state = "WI",
  year = 2019,
  survey = "acs1"
)
```

```
## Getting data from the 2019 1-year ACS
```

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

```
print(wi_income_1yr)
```

```
## # A tibble: 23 × 5
##    GEOID NAME                           variable   estimate   moe
##    <chr> <chr>                          <chr>         <dbl> <dbl>
##  1 55009 Brown County, Wisconsin        B19013_001    64458  3713
##  2 55025 Dane County, Wisconsin         B19013_001    77504  3180
##  3 55027 Dodge County, Wisconsin        B19013_001    61300  2119
##  4 55035 Eau Claire County, Wisconsin   B19013_001    65662  3914
##  5 55039 Fond du Lac County, Wisconsin  B19013_001    65329  4646
##  6 55055 Jefferson County, Wisconsin    B19013_001    71708  5901
##  7 55059 Kenosha County, Wisconsin      B19013_001    65997  4897
##  8 55063 La Crosse County, Wisconsin    B19013_001    58921  3969
##  9 55071 Manitowoc County, Wisconsin    B19013_001    60785  3913
## 10 55073 Marathon County, Wisconsin     B19013_001    65851  2868
## # … with 13 more rows
```

```
nrow(wi_income_1yr)
```

```
## [1] 23
```

```
v16 <- load_variables(2016, "acs5", cache = TRUE)

#View(v16)
```

```
hhinc <- get_acs(
  geography = "state",
  table = "B19001",
  survey = "acs1",
  year = 2016
)
```

```
## Getting data from the 2016 1-year ACS
```

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

```
## Loading ACS1 variables for 2016 from table B19001. To cache this dataset for faster access to ACS tables in the future, run this function with `cache_table = TRUE`. You only need to do this once per ACS dataset.
```

```
print(hhinc)
```

```
## # A tibble: 884 × 5
##    GEOID NAME    variable   estimate   moe
##    <chr> <chr>   <chr>         <dbl> <dbl>
##  1 01    Alabama B19001_001  1852518 12189
##  2 01    Alabama B19001_002   176641  6328
##  3 01    Alabama B19001_003   120590  5347
##  4 01    Alabama B19001_004   117332  5956
##  5 01    Alabama B19001_005   108912  5308
##  6 01    Alabama B19001_006   102080  4740
##  7 01    Alabama B19001_007   103366  5246
##  8 01    Alabama B19001_008    91011  4699
##  9 01    Alabama B19001_009    86996  4418
## 10 01    Alabama B19001_010    74864  4210
## # … with 874 more rows
```

```
hhinc_wide <- get_acs(
  geography = "state",
  table = "B19001",
  survey = "acs1",
  year = 2016,
  output = "wide"
)
```

```
## Getting data from the 2016 1-year ACS
```

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

```
## Loading ACS1 variables for 2016 from table B19001. To cache this dataset for faster access to ACS tables in the future, run this function with `cache_table = TRUE`. You only need to do this once per ACS dataset.
```

```
print(hhinc_wide)
```

```
## # A tibble: 52 × 36
##    GEOID NAME    B1900…¹ B1900…² B1900…³ B1900…⁴ B1900…⁵ B1900…⁶ B1900…⁷ B1900…⁸
##    <chr> <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 28    Missis… 1091245    8803  113124    4835   87136    5004   71206    4058
##  2 29    Missou… 2372190   10844  160615    6705  122649    4654  123789    5201
##  3 30    Montana  416125    4426   26734    2183   24786    2391   22330    2391
##  4 31    Nebras…  747562    4452   45794    3116   33266    2466   31084    2533
##  5 32    Nevada  1055158    6433   68507    4886   42720    3071   53143    3653
##  6 33    New Ha…  520643    5191   20890    2566   15933    1908   18190    2315
##  7 34    New Je… 3194519   10274  170029    6836  118862    5855  123335    6065
##  8 35    New Me…  758364    6296   66983    4439   48930    3220   50025    4091
##  9 36    New Yo… 7209054   17665  543763   12132  352029    9607  322683    7756
## 10 37    North … 3882423   16063  282491    7816  228088    7916  209825    6844
## # … with 42 more rows, 26 more variables: B19001_005E <dbl>, B19001_005M <dbl>,
## #   B19001_006E <dbl>, B19001_006M <dbl>, B19001_007E <dbl>, B19001_007M <dbl>,
## #   B19001_008E <dbl>, B19001_008M <dbl>, B19001_009E <dbl>, B19001_009M <dbl>,
## #   B19001_010E <dbl>, B19001_010M <dbl>, B19001_011E <dbl>, B19001_011M <dbl>,
## #   B19001_012E <dbl>, B19001_012M <dbl>, B19001_013E <dbl>, B19001_013M <dbl>,
## #   B19001_014E <dbl>, B19001_014M <dbl>, B19001_015E <dbl>, B19001_015M <dbl>,
## #   B19001_016E <dbl>, B19001_016M <dbl>, B19001_017E <dbl>, …
```

```
cimarron_blocks <- get_decennial(
  geography = "block",
  variables = "H1_001N",
  state = "OK",
  county = "Cimarron",
  year = 2020,
  sumfile = "pl"
)
```

```
## Getting data from the 2020 decennial Census
```

```
## Using the PL 94-171 Redistricting Data summary file
```

```
print(cimarron_blocks)
```

```
## # A tibble: 1,269 × 4
##    GEOID           NAME                                              varia…¹ value
##    <chr>           <chr>                                             <chr>   <dbl>
##  1 400259501001984 Block 1984, Block Group 1, Census Tract 9501, … H1_001N     0
##  2 400259503001035 Block 1035, Block Group 1, Census Tract 9503, … H1_001N     0
##  3 400259503001068 Block 1068, Block Group 1, Census Tract 9503, … H1_001N     5
##  4 400259503001146 Block 1146, Block Group 1, Census Tract 9503, … H1_001N     0
##  5 400259503001197 Block 1197, Block Group 1, Census Tract 9503, … H1_001N     7
##  6 400259503001218 Block 1218, Block Group 1, Census Tract 9503, … H1_001N     2
##  7 400259501001067 Block 1067, Block Group 1, Census Tract 9501, … H1_001N     0
##  8 400259501001118 Block 1118, Block Group 1, Census Tract 9501, … H1_001N     0
##  9 400259501001141 Block 1141, Block Group 1, Census Tract 9501, … H1_001N     0
## 10 400259501001223 Block 1223, Block Group 1, Census Tract 9501, … H1_001N     0
## # … with 1,259 more rows, and abbreviated variable name ¹variable
```

```
ga <- get_acs(
  geography = "county",
  state = "Georgia",
  variables = c(medinc = "B19013_001",
                medage = "B01002_001"),
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
print(ga)
```

```
## # A tibble: 318 × 5
##    GEOID NAME                      variable estimate    moe
##    <chr> <chr>                     <chr>       <dbl>  <dbl>
##  1 13001 Appling County, Georgia   medage       39.9    1.7
##  2 13001 Appling County, Georgia   medinc    37924   4761
##  3 13003 Atkinson County, Georgia  medage       35.9    1.5
##  4 13003 Atkinson County, Georgia  medinc    35703   5493
##  5 13005 Bacon County, Georgia     medage       36.5    1
##  6 13005 Bacon County, Georgia     medinc    36692   3774
##  7 13007 Baker County, Georgia     medage       52.2    4.8
##  8 13007 Baker County, Georgia     medinc    34034   9879
##  9 13009 Baldwin County, Georgia   medage       35.8    0.5
## 10 13009 Baldwin County, Georgia   medinc    46250   4707
## # … with 308 more rows
```

```
ga_wide <- get_acs(
  geography = "county",
  state = "Georgia",
  variables = c(medinc = "B19013_001",
                medage = "B01002_001"),
  output = "wide",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
print(ga_wide)
```

```
## # A tibble: 159 × 6
##    GEOID NAME                      medincE medincM medageE medageM
##    <chr> <chr>                       <dbl>   <dbl>   <dbl>   <dbl>
##  1 13001 Appling County, Georgia     37924    4761    39.9     1.7
##  2 13003 Atkinson County, Georgia    35703    5493    35.9     1.5
##  3 13005 Bacon County, Georgia       36692    3774    36.5     1
##  4 13007 Baker County, Georgia       34034    9879    52.2     4.8
##  5 13011 Banks County, Georgia       50912    4278    41.5     1.1
##  6 13013 Barrow County, Georgia      62990    2562    36       0.3
##  7 13017 Ben Hill County, Georgia    32077    4008    39.5     1.4
##  8 13021 Bibb County, Georgia        41317    1220    36.3     0.3
##  9 13023 Bleckley County, Georgia    46992    6279    36       1.5
## 10 13027 Brooks County, Georgia      37516    4438    43.6     0.9
## # … with 149 more rows
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.0      ✓ purrr   1.0.1
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.3.0      ✓ stringr 1.5.0
## ✓ readr   2.1.3      ✓ forcats 1.0.0
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
## Warning: package 'tidyr' was built under R version 4.2.2
```

```
## Warning: package 'readr' was built under R version 4.2.2
```

```
## Warning: package 'purrr' was built under R version 4.2.2
```

```
## Warning: package 'stringr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

```
## ── Conflicts ─────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
queens_components <- get_estimates(
  geography = "county",
  product = "components",
  state = "NY",
  county = "Queens",
  year = 2019
)

print(queens_components)
```

```
## # A tibble: 12 × 4
##    NAME                    GEOID variable             value
##    <chr>                   <chr> <chr>                <dbl>
##  1 Queens County, New York 36081 BIRTHS             27453
##  2 Queens County, New York 36081 DEATHS             16380
##  3 Queens County, New York 36081 DOMESTICMIG       -41789
##  4 Queens County, New York 36081 INTERNATIONALMIG    9883
##  5 Queens County, New York 36081 NATURALINC         11073
##  6 Queens County, New York 36081 NETMIG            -31906
##  7 Queens County, New York 36081 RBIRTH                12.1
##  8 Queens County, New York 36081 RDEATH                 7.23
##  9 Queens County, New York 36081 RDOMESTICMIG         -18.5
## 10 Queens County, New York 36081 RINTERNATIONALMIG      4.36
## 11 Queens County, New York 36081 RNATURALINC            4.89
## 12 Queens County, New York 36081 RNETMIG              -14.1
```

```
louisiana_sex_hisp <- get_estimates(
  geography = "state",
  product = "characteristics",
  breakdown = c("SEX", "HISP"),
  breakdown_labels = TRUE,
  state = "LA",
  year = 2019
)

print(louisiana_sex_hisp)
```

```
## # A tibble: 9 × 5
##   GEOID NAME        value SEX        HISP
##   <chr> <chr>       <dbl> <chr>      <chr>
## 1 22    Louisiana 4648794 Both sexes Both Hispanic Origins
## 2 22    Louisiana 4401822 Both sexes Non-Hispanic
## 3 22    Louisiana  246972 Both sexes Hispanic
## 4 22    Louisiana 2267050 Male       Both Hispanic Origins
## 5 22    Louisiana 2135979 Male       Non-Hispanic
## 6 22    Louisiana  131071 Male       Hispanic
## 7 22    Louisiana 2381744 Female     Both Hispanic Origins
## 8 22    Louisiana 2265843 Female     Non-Hispanic
## 9 22    Louisiana  115901 Female     Hispanic
```

```
honolulu_migration <- get_flows(
  geography = "county",
  state = "HI",
  county = "Honolulu",
  year = 2019
)

print(honolulu_migration)
```

```
## # A tibble: 3,156 × 7
##    GEOID1 GEOID2 FULL1_NAME              FULL2_NAME      variable estimate   moe
##    <chr>  <chr>  <chr>                   <chr>           <chr>       <dbl> <dbl>
##  1 15003  <NA>   Honolulu County, Hawaii Africa          MOVEDIN       152   156
##  2 15003  <NA>   Honolulu County, Hawaii Africa          MOVEDOUT       NA    NA
##  3 15003  <NA>   Honolulu County, Hawaii Africa          MOVEDNET       NA    NA
##  4 15003  <NA>   Honolulu County, Hawaii Asia            MOVEDIN      7680   884
##  5 15003  <NA>   Honolulu County, Hawaii Asia            MOVEDOUT       NA    NA
##  6 15003  <NA>   Honolulu County, Hawaii Asia            MOVEDNET       NA    NA
##  7 15003  <NA>   Honolulu County, Hawaii Central America MOVEDIN       192   100
##  8 15003  <NA>   Honolulu County, Hawaii Central America MOVEDOUT       NA    NA
##  9 15003  <NA>   Honolulu County, Hawaii Central America MOVEDNET       NA    NA
## 10 15003  <NA>   Honolulu County, Hawaii Caribbean       MOVEDIN        97    78
## # … with 3,146 more rows
```

```
cbsa_bachelors <- get_acs(
  geography = "cbsa",
  variables = "DP02_0068P",
  year = 2019,
  show_call = TRUE
)
```

```
## Getting data from the 2015-2019 5-year ACS
```

```
## Using the ACS Data Profile
```

```
## Census API call: https://api.census.gov/data/2019/acs/acs5/profile?get=DP02_0068PE%2CDP02_0068PM%2CNAME&for=metropolitan%20statistical%20area%2Fmicropolitan%20statistical%20area%3A%2A
```

```
print(cbsa_bachelors)
```

```
## # A tibble: 938 × 5
##    GEOID NAME                              variable   estimate   moe
##    <chr> <chr>                             <chr>         <dbl> <dbl>
##  1 10100 Aberdeen, SD Micro Area           DP02_0068P     29.1   2.1
##  2 10140 Aberdeen, WA Micro Area           DP02_0068P     16.5   1.3
##  3 10180 Abilene, TX Metro Area            DP02_0068P     23     1
##  4 10220 Ada, OK Micro Area                DP02_0068P     28.2   1.5
##  5 10300 Adrian, MI Micro Area             DP02_0068P     21     0.9
##  6 10380 Aguadilla-Isabela, PR Metro Area  DP02_0068P     NA    NA
##  7 10420 Akron, OH Metro Area              DP02_0068P     31.7   0.5
##  8 10460 Alamogordo, NM Micro Area         DP02_0068P     17.5   1.5
##  9 10500 Albany, GA Metro Area             DP02_0068P     20     0.9
## 10 10540 Albany-Lebanon, OR Metro Area     DP02_0068P     19.3   1.2
## # … with 928 more rows
```

Wrangling Census data with tidyverse tools

Ram Mandava

2023-02-17

Exploring Census data with tidyverse tools

```
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.2.2
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.0      ✓ purrr   1.0.1
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.3.0      ✓ stringr 1.5.0
## ✓ readr   2.1.3      ✓ forcats 1.0.0
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
## Warning: package 'tidyr' was built under R version 4.2.2
```

```
## Warning: package 'readr' was built under R version 4.2.2
```

```
## Warning: package 'purrr' was built under R version 4.2.2
```

```
## Warning: package 'stringr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
median_age <- get_acs(
  geography = "county",
  variables = "B01002_001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
arrange(median_age, estimate)
```

```
## # A tibble: 3,221 × 5
##    GEOID NAME                          variable   estimate   moe
##    <chr> <chr>                         <chr>         <dbl> <dbl>
##  1 35011 De Baca County, New Mexico    B01002_001     22.2   6.9
##  2 51678 Lexington city, Virginia      B01002_001     22.2   0.8
##  3 16065 Madison County, Idaho         B01002_001     23.5   0.2
##  4 46121 Todd County, South Dakota     B01002_001     23.6   0.6
##  5 51750 Radford city, Virginia        B01002_001     23.7   0.6
##  6 13053 Chattahoochee County, Georgia B01002_001     24     0.7
##  7 02158 Kusilvak Census Area, Alaska  B01002_001     24.1   0.2
##  8 49049 Utah County, Utah             B01002_001     25     0.1
##  9 46027 Clay County, South Dakota     B01002_001     25.2   0.5
## 10 53075 Whitman County, Washington    B01002_001     25.2   0.2
## # … with 3,211 more rows
```

```
arrange(median_age, desc(estimate))
```

```
## # A tibble: 3,221 × 5
##    GEOID NAME                              variable   estimate   moe
##    <chr> <chr>                             <chr>         <dbl> <dbl>
##  1 12119 Sumter County, Florida            B01002_001     68     0.3
##  2 48301 Loving County, Texas              B01002_001     62.2  37.8
##  3 48243 Jeff Davis County, Texas          B01002_001     61.3  36.8
##  4 08027 Custer County, Colorado           B01002_001     60.1   3.4
##  5 12015 Charlotte County, Florida         B01002_001     59.5   0.2
##  6 51091 Highland County, Virginia         B01002_001     59.5   4.8
##  7 35003 Catron County, New Mexico         B01002_001     59.4   2.6
##  8 51133 Northumberland County, Virginia   B01002_001     59.3   0.7
##  9 26131 Ontonagon County, Michigan        B01002_001     59.1   0.5
## 10 48443 Terrell County, Texas             B01002_001     59.1   9.4
## # … with 3,211 more rows
```

```
filter(median_age, estimate >= 50)
```

```
## # A tibble: 218 × 5
##    GEOID NAME                                variable   estimate   moe
##    <chr> <chr>                               <chr>         <dbl> <dbl>
##  1 02105 Hoonah-Angoon Census Area, Alaska B01002_001     52.1   2.9
##  2 04007 Gila County, Arizona              B01002_001     50.4   0.2
##  3 04012 La Paz County, Arizona            B01002_001     57.4   0.6
##  4 04015 Mohave County, Arizona            B01002_001     52.3   0.2
##  5 04025 Yavapai County, Arizona           B01002_001     54.1   0.2
##  6 05005 Baxter County, Arkansas           B01002_001     52.3   0.5
##  7 05089 Marion County, Arkansas           B01002_001     52.1   0.8
##  8 05097 Montgomery County, Arkansas       B01002_001     50.6   0.8
##  9 05137 Stone County, Arkansas            B01002_001     50     0.5
## 10 06009 Calaveras County, California      B01002_001     52.8   0.6
## # … with 208 more rows
```

```
separate(
  median_age,
  NAME,
  into = c("county", "state"),
  sep = ", "
)
```

```
## # A tibble: 3,221 × 6
##    GEOID county          state   variable   estimate   moe
##    <chr> <chr>           <chr>   <chr>         <dbl> <dbl>
##  1 01001 Autauga County  Alabama B01002_001     38.6   0.6
##  2 01003 Baldwin County  Alabama B01002_001     43.2   0.4
##  3 01005 Barbour County  Alabama B01002_001     40.1   0.6
##  4 01007 Bibb County     Alabama B01002_001     39.9   1.2
##  5 01009 Blount County   Alabama B01002_001     41     0.5
##  6 01011 Bullock County  Alabama B01002_001     39.7   1.9
##  7 01013 Butler County   Alabama B01002_001     41.2   0.6
##  8 01015 Calhoun County  Alabama B01002_001     39.5   0.4
##  9 01017 Chambers County Alabama B01002_001     41.9   0.7
## 10 01019 Cherokee County Alabama B01002_001     46.8   0.5
## # … with 3,211 more rows
```

```
race_vars <- c(
  White = "B03002_003",
  Black = "B03002_004",
  Native = "B03002_005",
  Asian = "B03002_006",
  HIPI = "B03002_007",
  Hispanic = "B03002_012"
)

az_race <- get_acs(
  geography = "county",
  state = "AZ",
  variables = race_vars,
  summary_var = "B03002_001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
az_race_percent <- az_race %>%
  mutate(percent = 100 * (estimate / summary_est)) %>%
  select(NAME, variable, percent)

largest_group <- az_race_percent %>%
  group_by(NAME) %>%
  filter(percent == max(percent))

az_race_percent %>%
  group_by(variable) %>%
  summarize(median_pct = median(percent))
```

```
## # A tibble: 6 × 2
##   variable median_pct
##   <chr>         <dbl>
## 1 Asian         0.992
## 2 Black         1.29
## 3 HIPI          0.107
## 4 Hispanic     30.5
## 5 Native        3.63
## 6 White        53.8
```

```
mn_hh_income <- get_acs(
  geography = "county",
  table = "B19001",
  state = "MN",
  year = 2016
)
```

```
## Getting data from the 2012-2016 5-year ACS
```

```
mn_hh_income_recode <- mn_hh_income %>%
  filter(variable != "B19001_001") %>%
  mutate(incgroup = case_when(
    variable < "B19001_008" ~ "below35k",
    variable < "B19001_013" ~ "bw35kand75k",
    TRUE ~ "above75k"
  ))

mn_group_sums <- mn_hh_income_recode %>%
  group_by(GEOID, incgroup) %>%
  summarize(estimate = sum(estimate))
```

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

```
oglala_lakota_age <- get_acs(
  geography = "county",
  state = "SD",
  county = "Oglala Lakota",
  table = "B01001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## Loading ACS5 variables for 2020 from table B01001. To cache this dataset for faster access to
ACS tables in the future, run this function with `cache_table = TRUE`. You only need to do this
once per ACS dataset.
```

```
oglala_lakota_age_10 <- get_acs(
  geography = "county",
  state = "SD",
  county = "Shannon",
  table = "B01001",
  year = 2010
)
```

```
## Getting data from the 2006-2010 5-year ACS
```

```
## Loading ACS5 variables for 2010 from table B01001. To cache this dataset for faster access to
ACS tables in the future, run this function with `cache_table = TRUE`. You only need to do this
once per ACS dataset.
```

```
co_college19 <- get_acs(
  geography = "county",
  variables = "DP02_0068P",
  state = "CO",
  survey = "acs1",
  year = 2019
)
```

```
## Getting data from the 2019 1-year ACS
```

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

```
## Using the ACS Data Profile
```

```
co_college18 <- get_acs(
  geography = "county",
  variables = "DP02_0068P",
  state = "CO",
  survey = "acs1",
  year = 2018
)
```

```
## Getting data from the 2018 1-year ACS
```

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

```
## Using the ACS Data Profile
```

```
ak_income_compare <- get_acs(
  geography = "county",
  variables = c(
    income15 = "CP03_2015_062",
    income20 = "CP03_2020_062"
  ),
  state = "AK",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## Using the ACS Comparison Profile
```

```
college_vars <- c("B15002_015",
                  "B15002_016",
                  "B15002_017",
                  "B15002_018",
                  "B15002_032",
                  "B15002_033",
                  "B15002_034",
                  "B15002_035")

years <- 2010:2019
names(years) <- years

college_by_year <- map_dfr(years, ~{
  get_acs(
    geography = "county",
    variables = college_vars,
    state = "CO",
    summary_var = "B15002_001",
    survey = "acs1",
    year = .x
  )
}, .id = "year")
```

```
## Getting data from the 2010 1-year ACS
```

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

```
## Getting data from the 2011 1-year ACS
```

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

```
## 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.
```

```
## Getting data from the 2017 1-year ACS
```

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

```
## Getting data from the 2018 1-year ACS
```

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

```
## Getting data from the 2019 1-year ACS
```

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

```
college_by_year %>%
  arrange(NAME, variable, year)
```

```
## # A tibble: 920 × 8
##    year  GEOID NAME                   variable   estimate   moe summar…¹ summa…²
##    <chr> <chr> <chr>                  <chr>         <dbl> <dbl>    <dbl>   <dbl>
##  1 2010  08001 Adams County, Colorado B15002_015    20501  1983   275849     790
##  2 2011  08001 Adams County, Colorado B15002_015    21233  2124   281231     865
##  3 2012  08001 Adams County, Colorado B15002_015    19238  2020   287924     693
##  4 2013  08001 Adams County, Colorado B15002_015    23818  2445   295122     673
##  5 2014  08001 Adams County, Colorado B15002_015    20255  1928   304394     541
##  6 2015  08001 Adams County, Colorado B15002_015    22962  2018   312281     705
##  7 2016  08001 Adams County, Colorado B15002_015    25744  2149   318077     525
##  8 2017  08001 Adams County, Colorado B15002_015    26159  2320   324185     562
##  9 2018  08001 Adams County, Colorado B15002_015    28113  2078   331247     955
## 10 2019  08001 Adams County, Colorado B15002_015    27552  2070   336931     705
## # … with 910 more rows, and abbreviated variable names ¹summary_est,
## #   ²summary_moe
```

```
percent_college_by_year <- college_by_year %>%
  group_by(NAME, year) %>%
  summarize(numerator = sum(estimate),
            denominator = first(summary_est)) %>%
  mutate(pct_college = 100 * (numerator / denominator)) %>%
  pivot_wider(id_cols = NAME,
              names_from = year,
              values_from = pct_college)
```

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

```
get_acs(
  geography = "county",
  state = "Rhode Island",
  variables = "B19013_001",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## # A tibble: 5 × 5
##   GEOID NAME                              variable   estimate   moe
##   <chr> <chr>                             <chr>         <dbl> <dbl>
## 1 44001 Bristol County, Rhode Island      B19013_001    85413  6122
## 2 44003 Kent County, Rhode Island         B19013_001    75857  2022
## 3 44005 Newport County, Rhode Island      B19013_001    84282  2629
## 4 44007 Providence County, Rhode Island   B19013_001    62323  1270
## 5 44009 Washington County, Rhode Island   B19013_001    86970  3651
```

```
get_acs(
  geography = "county",
  state = "Rhode Island",
  variables = "B19013_001",
  year = 2020,
  moe_level = 99
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
## # A tibble: 5 × 5
##   GEOID NAME                            variable   estimate   moe
##   <chr> <chr>                           <chr>         <dbl> <dbl>
## 1 44001 Bristol County, Rhode Island    B19013_001    85413 9527.
## 2 44003 Kent County, Rhode Island       B19013_001    75857 3147.
## 3 44005 Newport County, Rhode Island    B19013_001    84282 4091.
## 4 44007 Providence County, Rhode Island B19013_001    62323 1976.
## 5 44009 Washington County, Rhode Island B19013_001    86970 5682.
```

```
vars <- paste0("B01001_0", c(20:25, 44:49))

vars
```

```
##  [1] "B01001_020" "B01001_021" "B01001_022" "B01001_023" "B01001_024"
##  [6] "B01001_025" "B01001_044" "B01001_045" "B01001_046" "B01001_047"
## [11] "B01001_048" "B01001_049"
```

```
salt_lake <- get_acs(
  geography = "tract",
  variables = vars,
  state = "Utah",
  county = "Salt Lake",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
example_tract <- salt_lake %>%
  filter(GEOID == "49035100100")

example_tract %>%
  select(-NAME)
```

```
## # A tibble: 12 × 4
##    GEOID       variable   estimate   moe
##    <chr>       <chr>         <dbl> <dbl>
##  1 49035100100 B01001_020       11    13
##  2 49035100100 B01001_021       25    18
##  3 49035100100 B01001_022        7    10
##  4 49035100100 B01001_023        4     7
##  5 49035100100 B01001_024        0    12
##  6 49035100100 B01001_025       17    20
##  7 49035100100 B01001_044        0    12
##  8 49035100100 B01001_045        4     7
##  9 49035100100 B01001_046       21    17
## 10 49035100100 B01001_047      123   168
## 11 49035100100 B01001_048       17    21
## 12 49035100100 B01001_049        0    12
```

```
moe_prop(25, 100, 5, 3)
```

```
## [1] 0.0494343
```

```
salt_lake_grouped <- salt_lake %>%
  mutate(sex = case_when(
    str_sub(variable, start = -2) < "26" ~ "Male",
    TRUE ~ "Female"
  )) %>%
  group_by(GEOID, sex) %>%
  summarize(sum_est = sum(estimate),
            sum_moe = moe_sum(moe, estimate))
```

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

Exploring US Census data with visualization

Ram Mandava

2023-02-17

Basic Census visualization with ggplot2

```
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.2.2
```

```
ga_wide <- get_acs(
  geography = "county",
  state = "Georgia",
  variables = c(medinc = "B19013_001",
                medage = "B01002_001"),
  output = "wide",
  year = 2020
)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.0      ✓ purrr   1.0.1
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.3.0      ✓ stringr 1.5.0
## ✓ readr   2.1.3      ✓ forcats 1.0.0
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
## Warning: package 'tidyr' was built under R version 4.2.2
```

```
## Warning: package 'readr' was built under R version 4.2.2
```

```
## Warning: package 'purrr' was built under R version 4.2.2
```

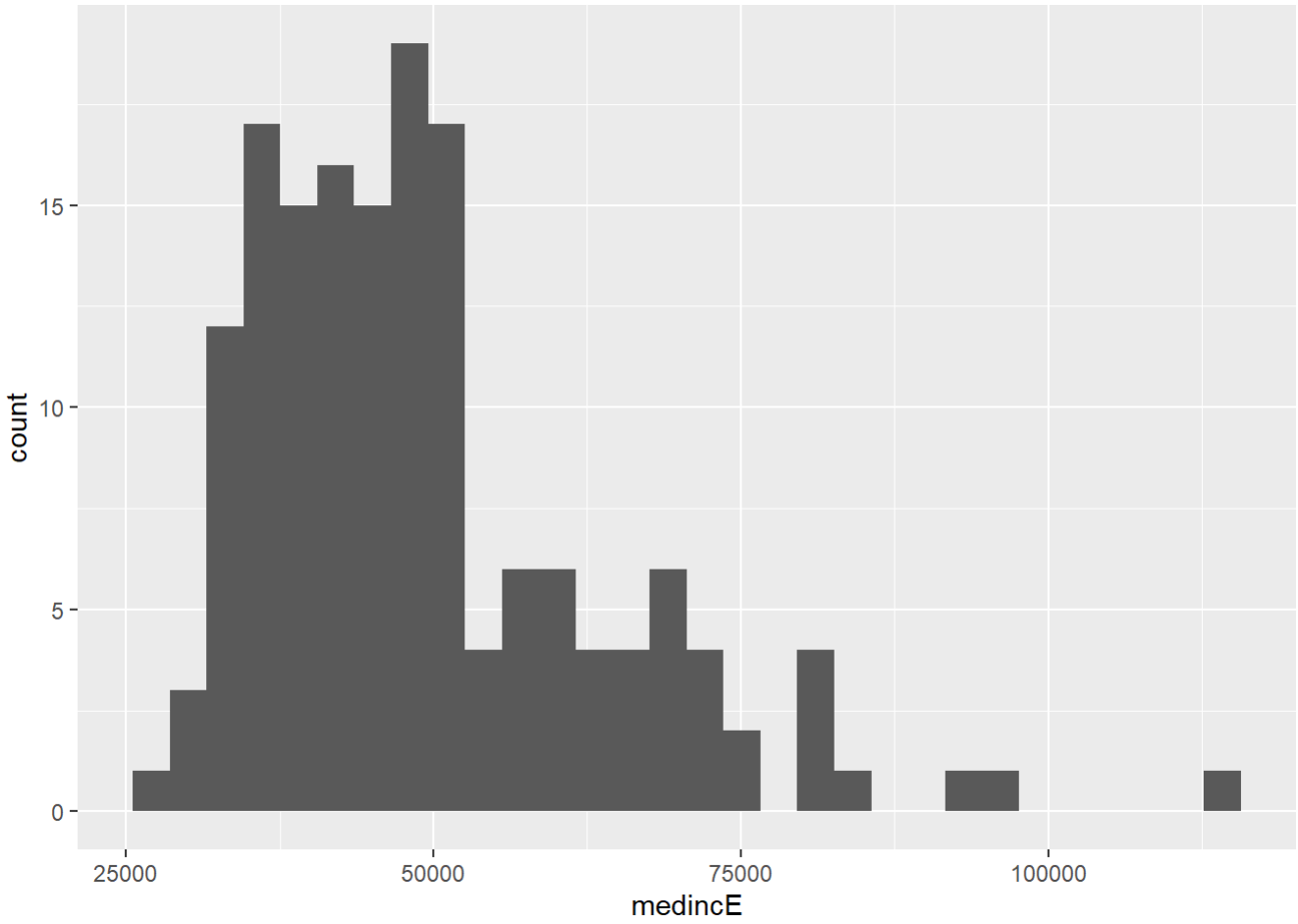
```
## Warning: package 'stringr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

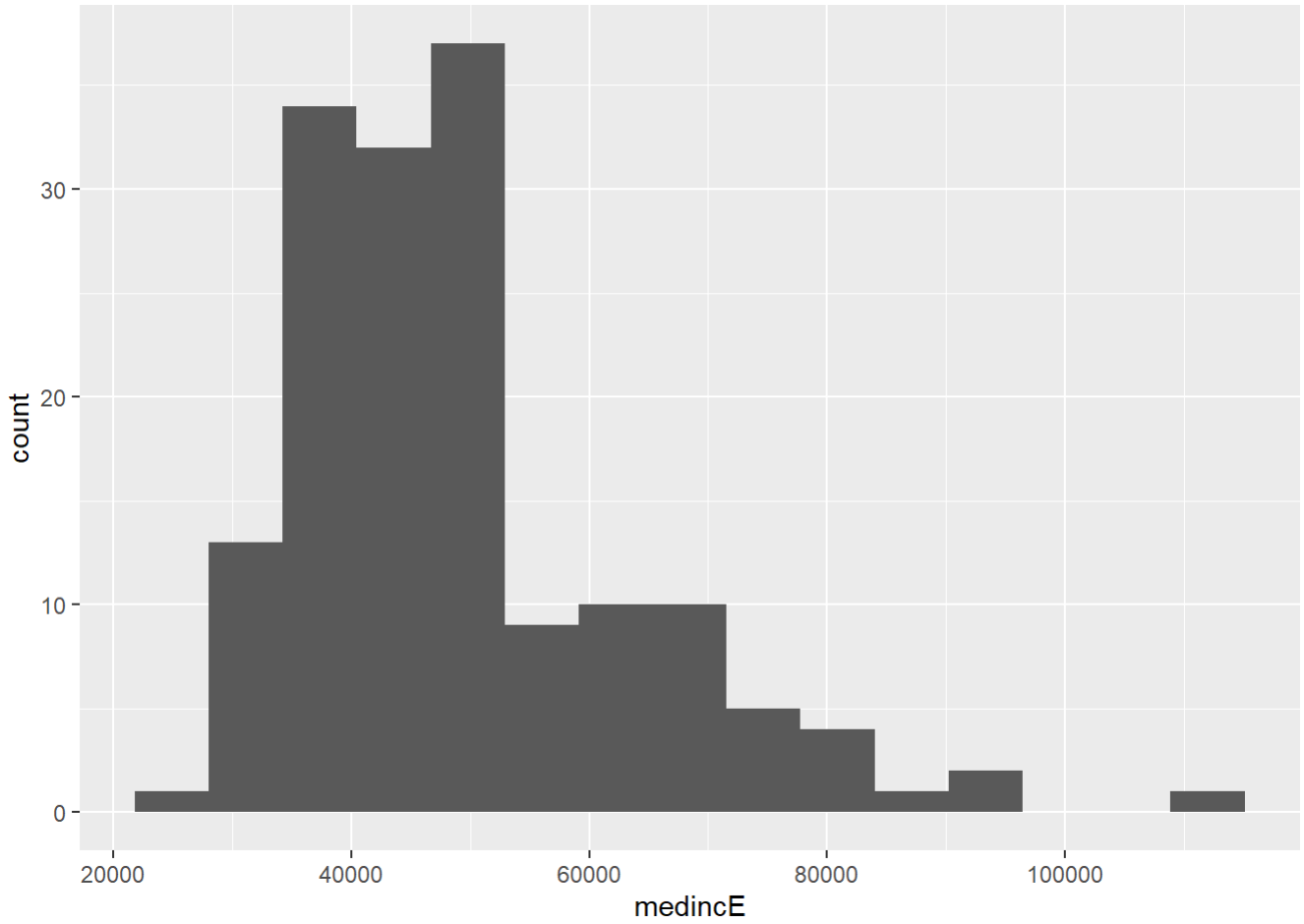
```
## ── Conflicts ─────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
options(scipen = 999)

ggplot(ga_wide, aes(x = medincE)) +
  geom_histogram()
```

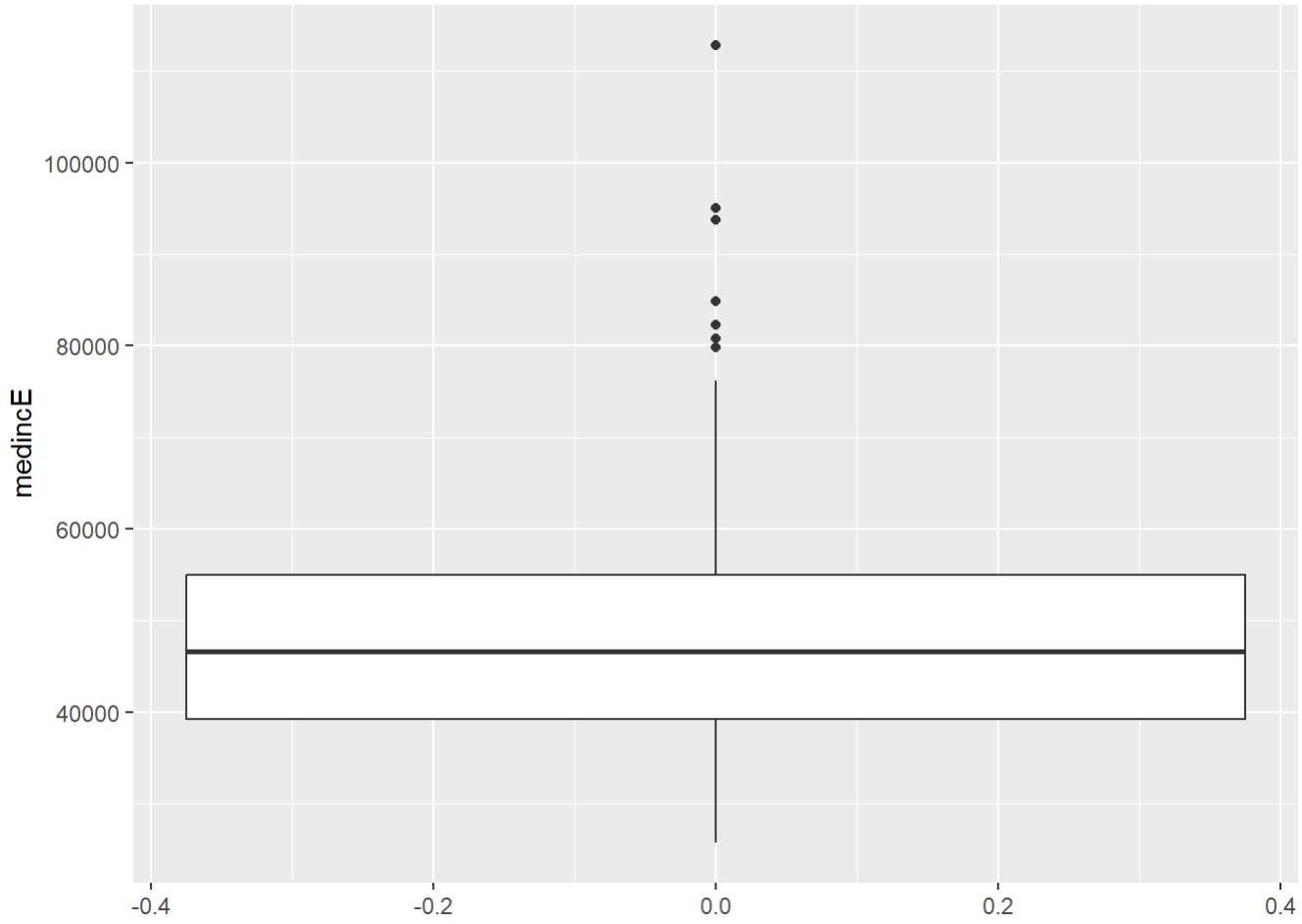
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



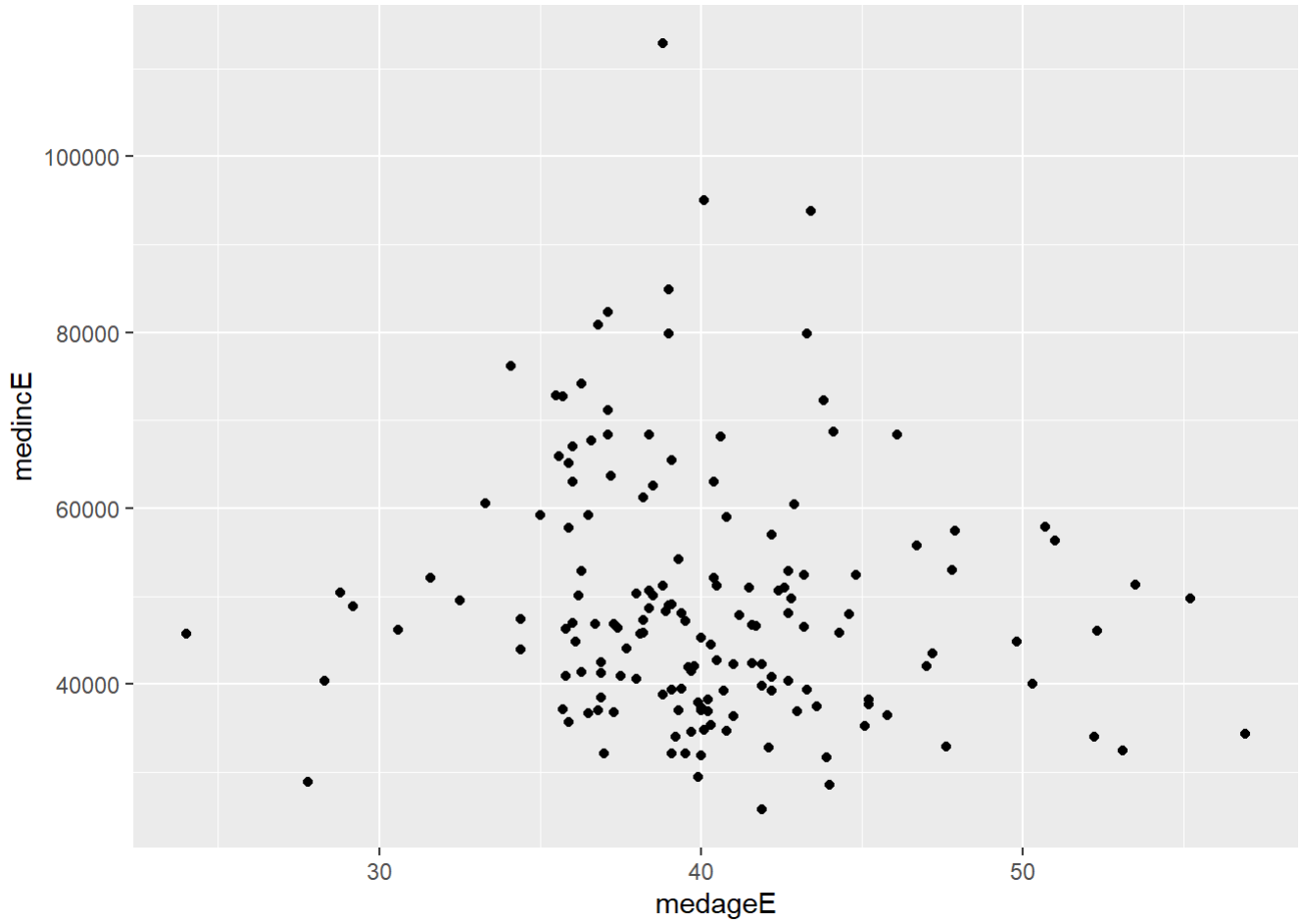

```
ggplot(ga_wide, aes(x = medincE)) +
  geom_histogram(bins = 15)
```




```
ggplot(ga_wide, aes(y = medincE)) +
  geom_boxplot()
```

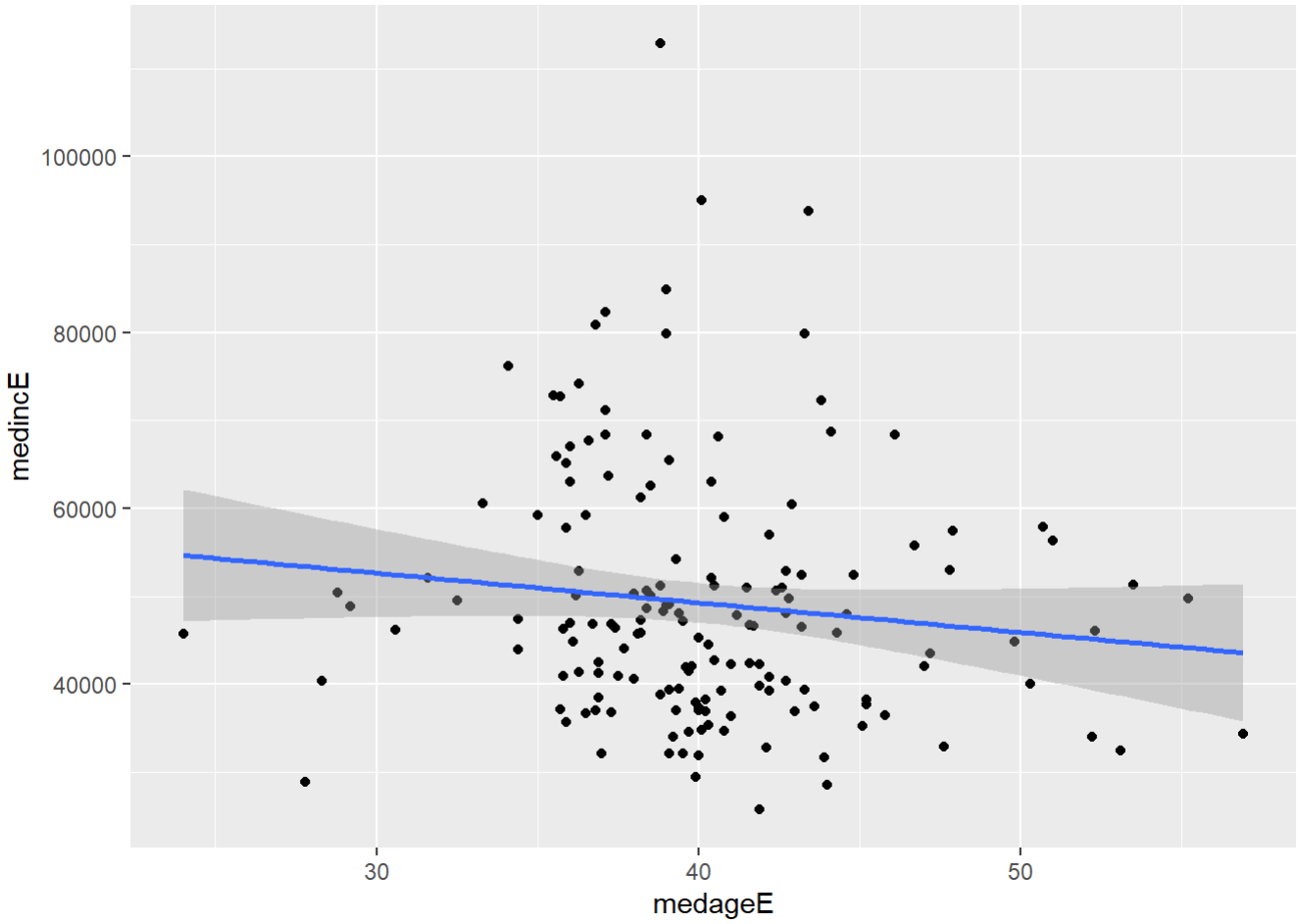



```
ggplot(ga_wide, aes(x = medageE, y = medincE)) +
  geom_point()
```




```
ggplot(ga_wide, aes(x = medageE, y = medincE)) +
  geom_point() +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



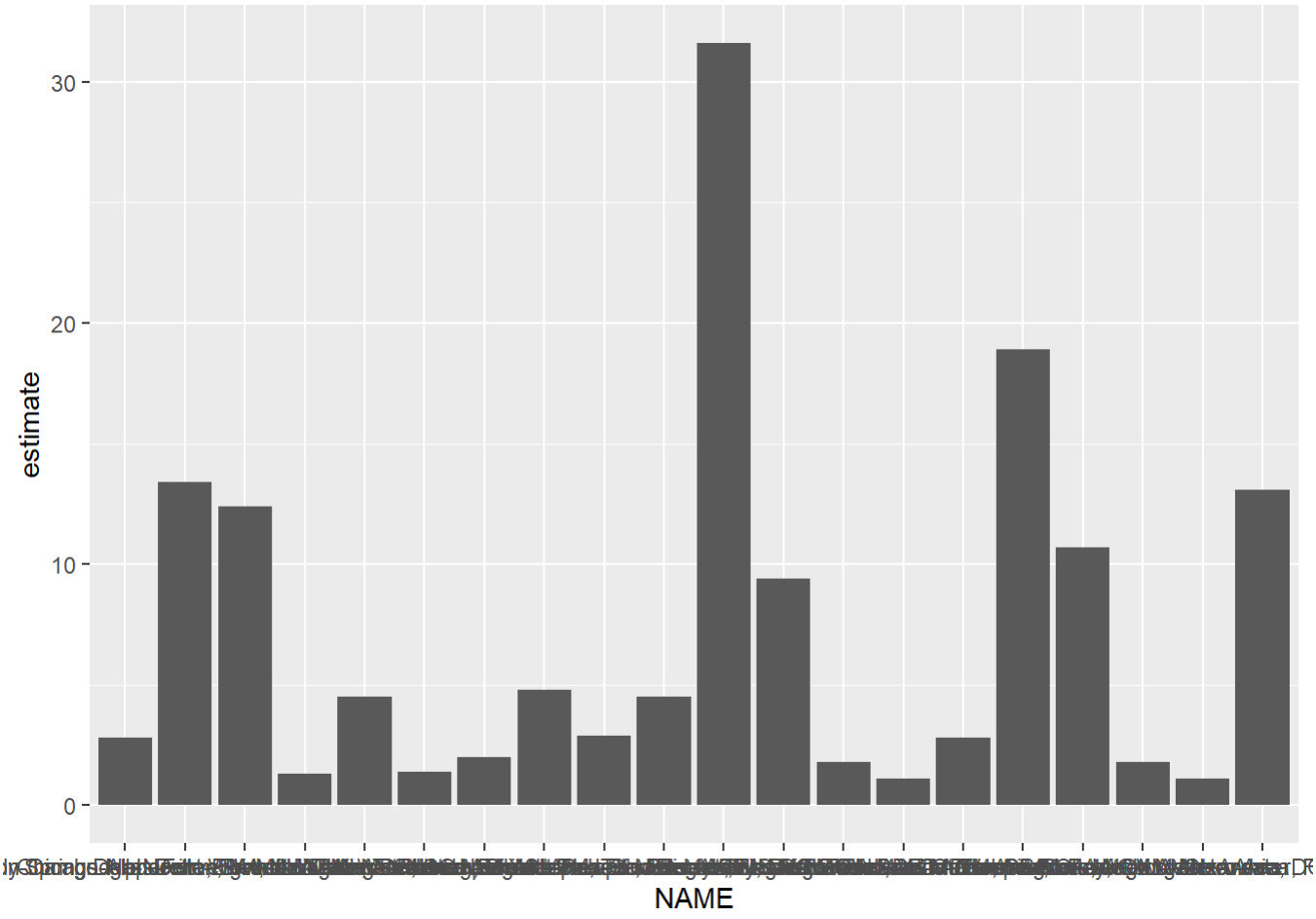

```
library(tidycensus)
library(tidyverse)

metros <-  get_acs(
  geography = "cbsa",
  variables = "DP03_0021P",
  summary_var = "B01003_001",
  survey = "acs1",
  year = 2019
) %>%
  slice_max(summary_est, n = 20)
```

```
## Getting data from the 2019 1-year ACS
```

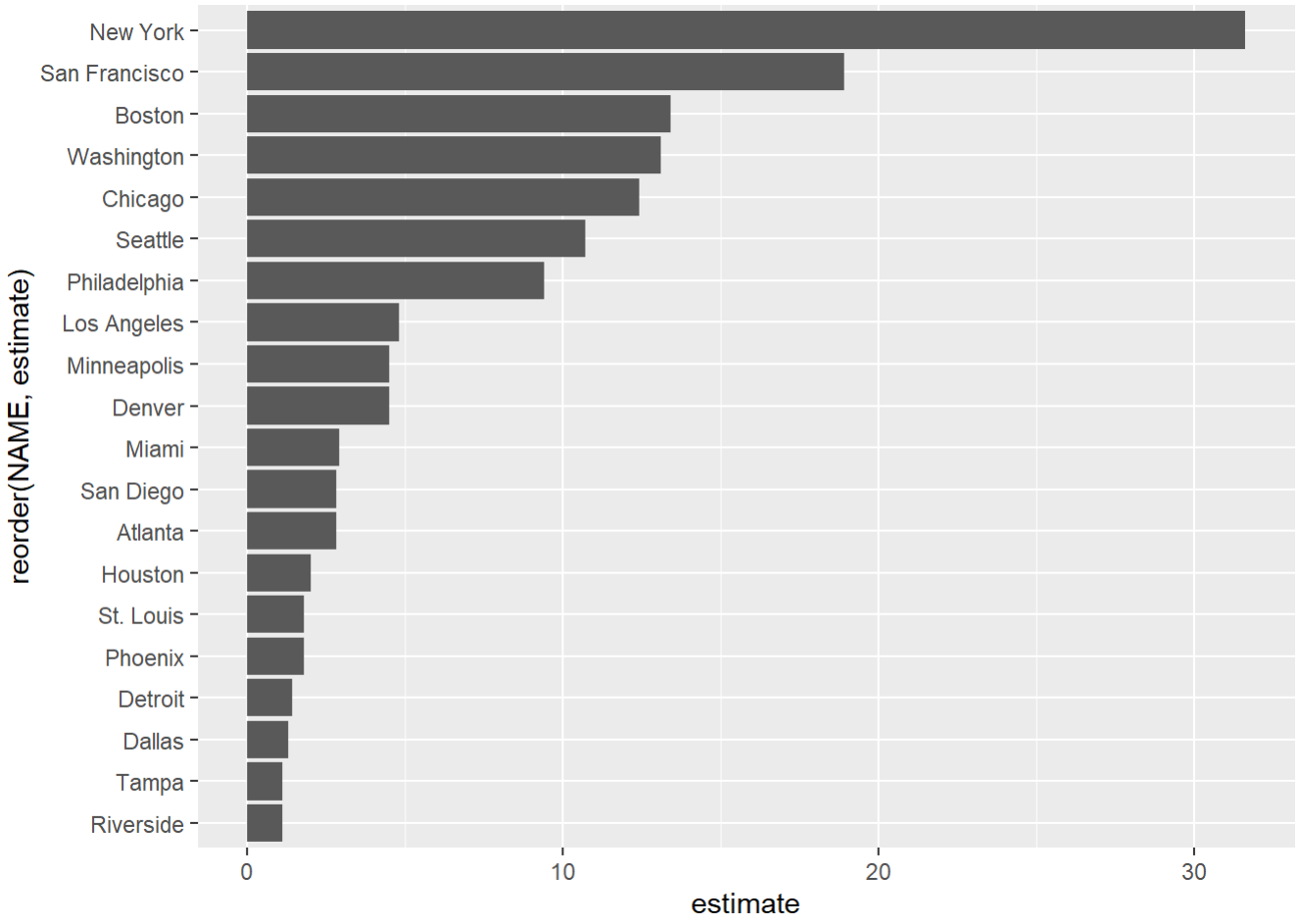
```
## The 1-year ACS provides data for geographies with populations of 65,000 and greater.
```

```
## Using the ACS Data Profile
```

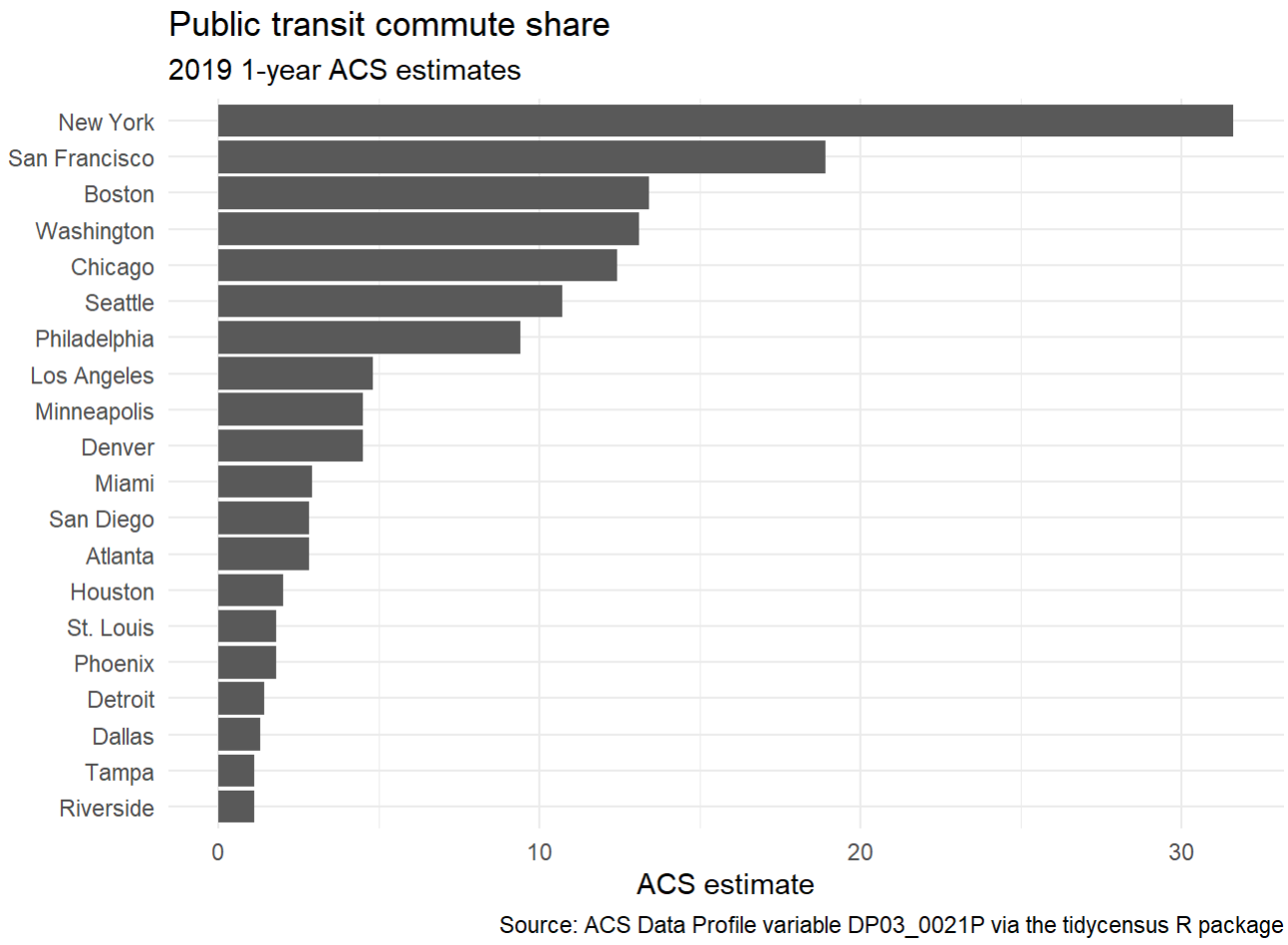
```
ggplot(metros, aes(x = NAME, y = estimate)) +
  geom_col()
```




```
metros %>%
  mutate(NAME = str_remove(NAME, "-.*$")) %>%
  mutate(NAME = str_remove(NAME, ",.*$")) %>%
  ggplot(aes(y = reorder(NAME, estimate), x = estimate)) +
  geom_col()
```




```
metros %>%
  mutate(NAME = str_remove(NAME, "-.*$")) %>%
  mutate(NAME = str_remove(NAME, ",.*$")) %>%
  ggplot(aes(y = reorder(NAME, estimate), x = estimate)) +
  geom_col() +
  theme_minimal() +
  labs(title = "Public transit commute share",
       subtitle = "2019 1-year ACS estimates",
       y = "",
       x = "ACS estimate",
       caption = "Source: ACS Data Profile variable DP03_0021P via the tidycensus R package")
```




```
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.2.2
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

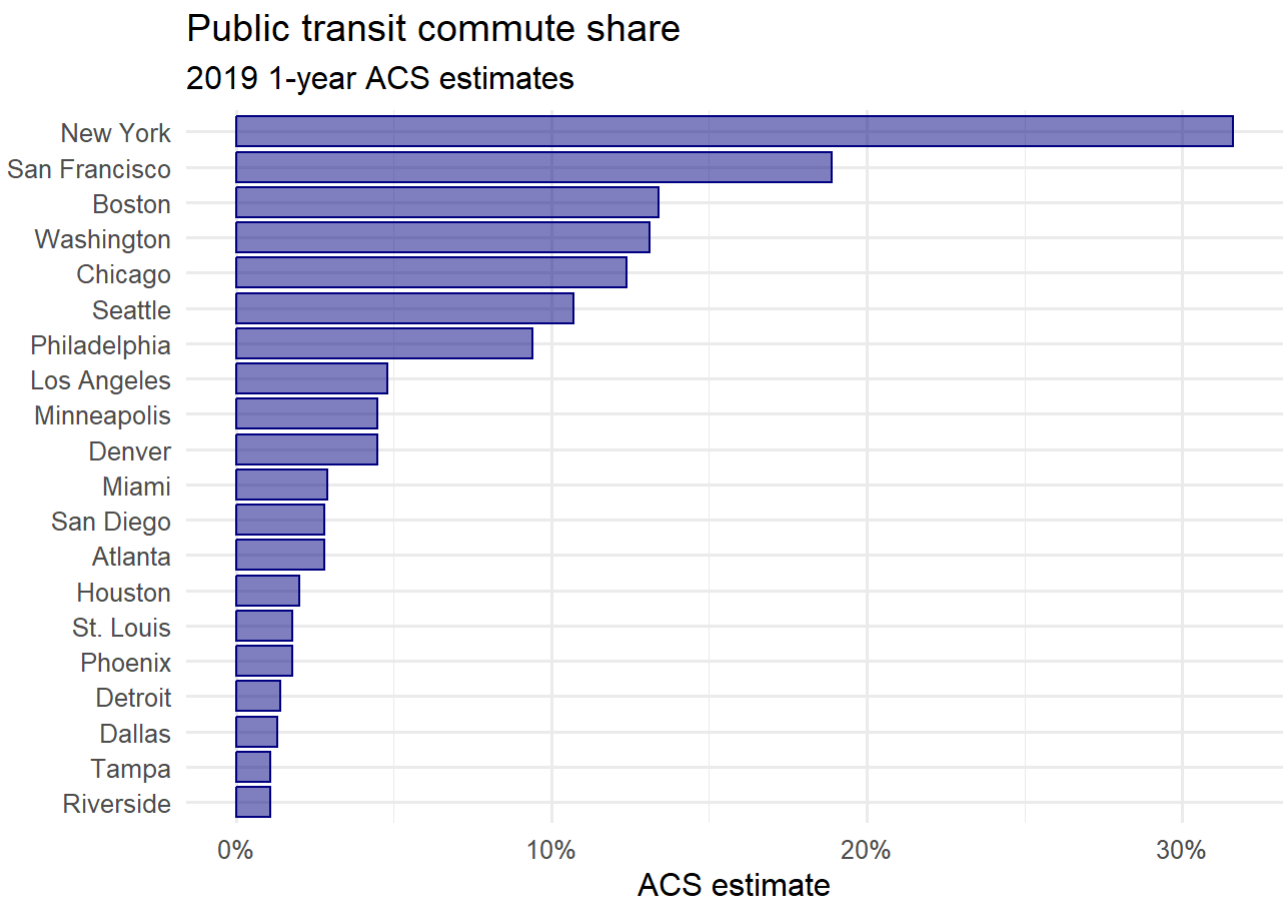
```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
metros %>%
  mutate(NAME = str_remove(NAME, "-.*$")) %>%
  mutate(NAME = str_remove(NAME, ",.*$")) %>%
  ggplot(aes(y = reorder(NAME, estimate), x = estimate)) +
  geom_col(color = "navy", fill = "navy",
           alpha = 0.5, width = 0.85) +
  theme_minimal(base_size = 12, base_family = "Verdana") +
  scale_x_continuous(labels = label_percent(scale = 1)) +
  labs(title = "Public transit commute share",
       subtitle = "2019 1-year ACS estimates",
       y = "",
       x = "ACS estimate",
       caption = "Source: ACS Data Profile variable DP03_0021P via the tidycensus R package")
```

```
## Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font family
## not found in Windows font database
```

```
## Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font family
## not found in Windows font database

## Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font family
## not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```




```
ggsave("metro_transit.png")
```

```
## Saving 7 x 5 in image
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```

```
ggsave(
  filename = "metro_transit.png",
  path = "~/images",
  width = 8,
  height = 5,
  units = "in",
  dpi = 300
)
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```

```
maine <- get_decennial(
  state = "Maine",
  geography = "county",
  variables = c(totalpop = "P1_001N"),
  year = 2020
) %>%
  arrange(desc(value))
```

```
## Getting data from the 2020 decennial Census
```

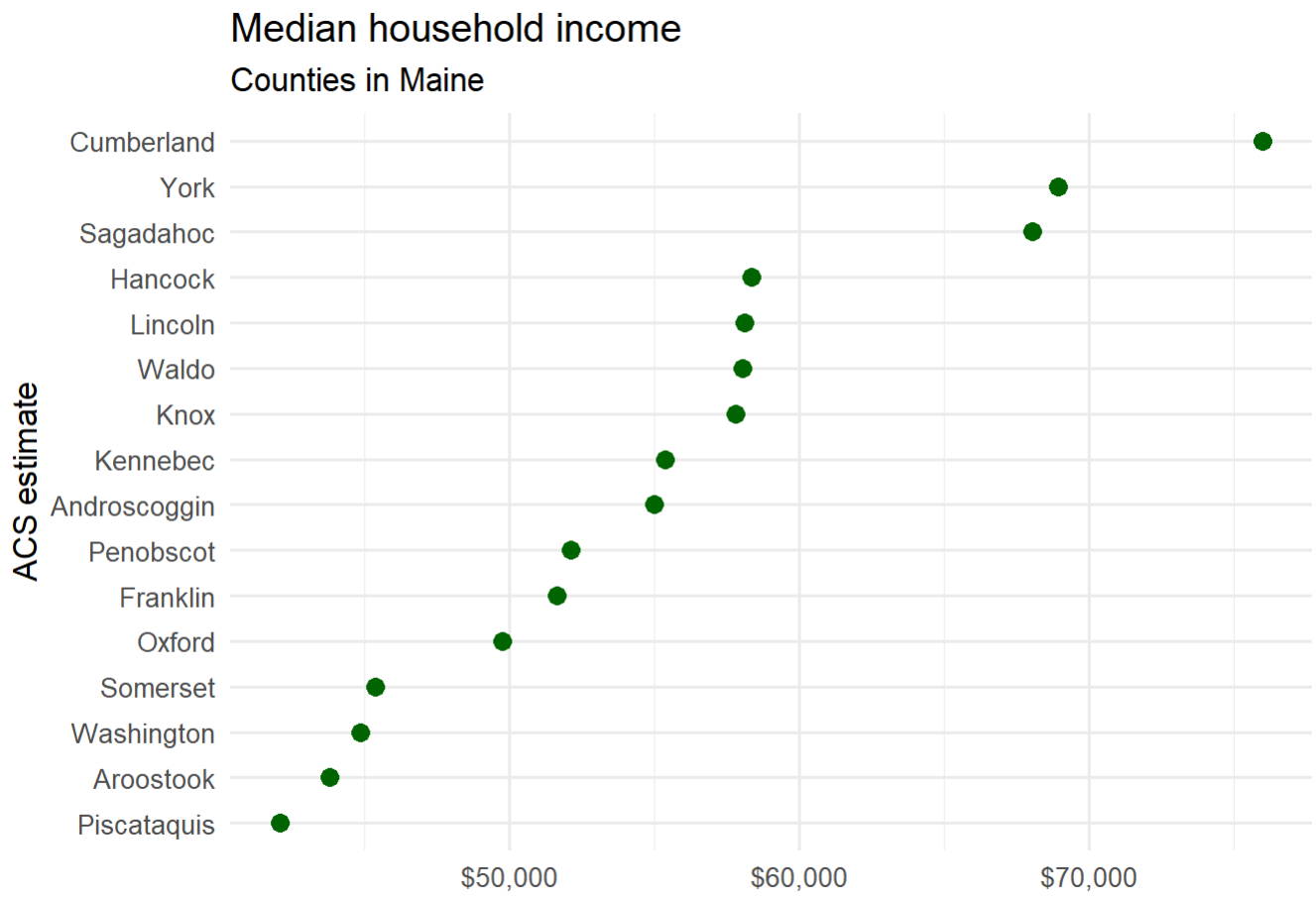
```
## Using the PL 94-171 Redistricting Data summary file
```

```
## Note: 2020 decennial Census data use differential privacy, a technique that
## introduces errors into data to preserve respondent confidentiality.
## ℹ Small counts should be interpreted with caution.
## ℹ See https://www.census.gov/library/fact-sheets/2021/protecting-the-confidentiality-of-the-2
020-census-redistricting-data.html for additional guidance.
## This message is displayed once per session.
```

```
maine_income <- get_acs(
  state = "Maine",
  geography = "county",
  variables = c(hhincome = "B19013_001"),
  year = 2020
) %>%
  mutate(NAME = str_remove(NAME, " County, Maine"))
```

```
## Getting data from the 2016-2020 5-year ACS
```

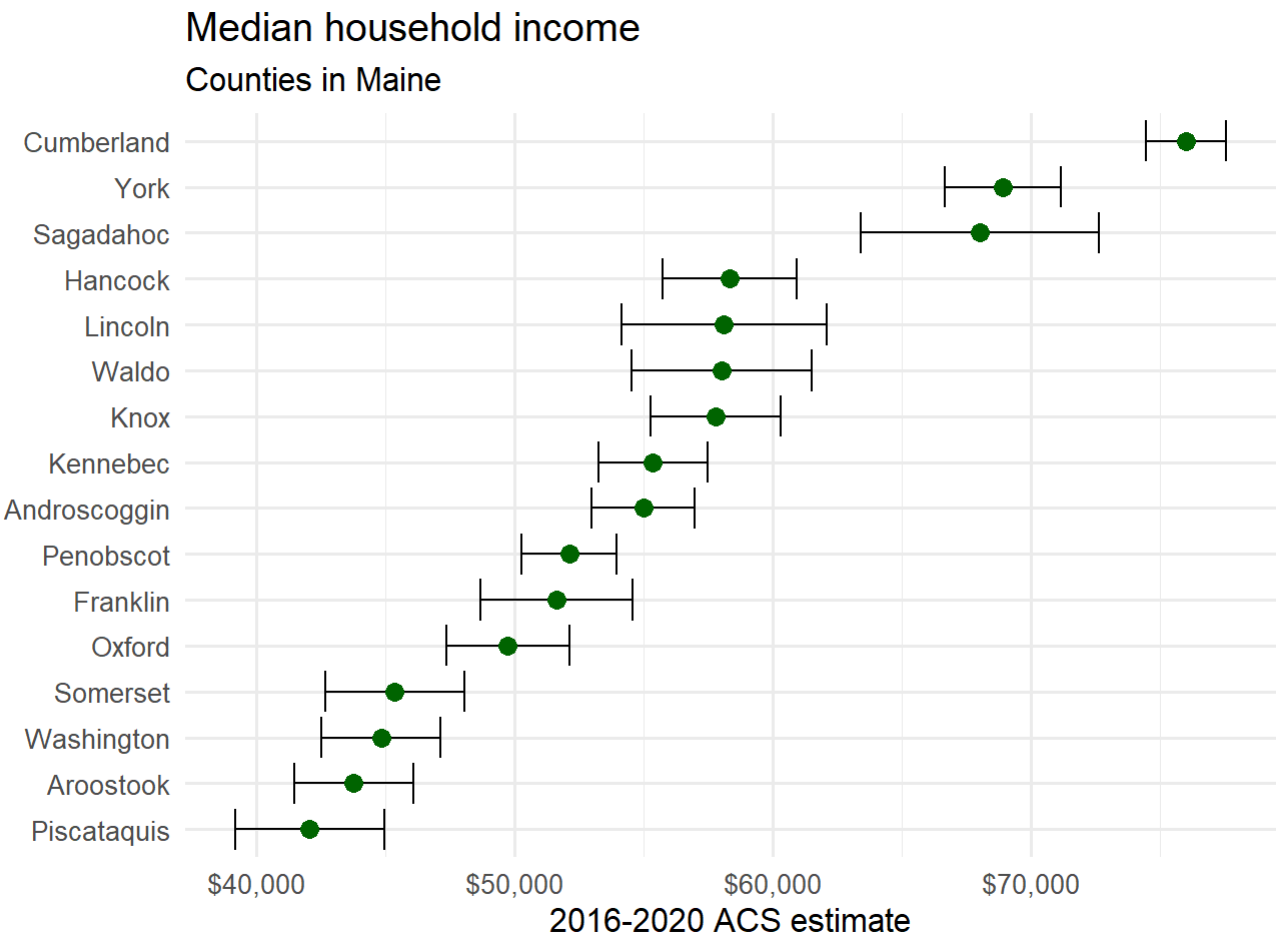
```
ggplot(maine_income, aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_point(size = 3, color = "darkgreen") +
  labs(title = "Median household income",
       subtitle = "Counties in Maine",
       x = "",
       y = "ACS estimate") +
  theme_minimal(base_size = 12.5) +
  scale_x_continuous(labels = label_dollar())
```




```
maine_income %>%
  arrange(desc(moe))
```

```
## # A tibble: 16 × 5
##    GEOID NAME         variable estimate   moe
##    <chr> <chr>        <chr>       <dbl> <dbl>
##  1 23023 Sagadahoc    hhincome    68039  4616
##  2 23015 Lincoln      hhincome    58125  3974
##  3 23027 Waldo        hhincome    58034  3482
##  4 23007 Franklin     hhincome    51630  2948
##  5 23021 Piscataquis  hhincome    42083  2883
##  6 23025 Somerset     hhincome    45382  2694
##  7 23009 Hancock      hhincome    58345  2593
##  8 23013 Knox         hhincome    57794  2528
##  9 23017 Oxford       hhincome    49761  2380
## 10 23003 Aroostook    hhincome    43791  2306
## 11 23029 Washington   hhincome    44847  2292
## 12 23031 York         hhincome    68932  2239
## 13 23011 Kennebec     hhincome    55368  2112
## 14 23001 Androscoggin hhincome    55002  2003
## 15 23019 Penobscot    hhincome    52128  1836
## 16 23005 Cumberland   hhincome    76014  1563
```

```
ggplot(maine_income, aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_errorbarh(aes(xmin = estimate - moe, xmax = estimate + moe)) +
  geom_point(size = 3, color = "darkgreen") +
  theme_minimal(base_size = 12.5) +
  labs(title = "Median household income",
       subtitle = "Counties in Maine",
       x = "2016-2020 ACS estimate",
       y = "") +
  scale_x_continuous(labels = label_dollar())
```




```
years <- 2005:2019
names(years) <- years

deschutes_value <- map_dfr(years, ~{
  get_acs(
    geography = "county",
    variables = "B25077_001",
    state = "OR",
    county = "Deschutes",
    year = .x,
    survey = "acs1"
  )
}, .id = "year")
```

```
## Getting data from the 2005 1-year ACS
```

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

```
## Getting data from the 2006 1-year ACS
```

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

```
## Getting data from the 2007 1-year ACS
```

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

```
## Getting data from the 2008 1-year ACS
```

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

```
## Getting data from the 2009 1-year ACS
```

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

```
## Getting data from the 2010 1-year ACS
```

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

```
## Getting data from the 2011 1-year ACS
```

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

```
## 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.
```

```
## Getting data from the 2017 1-year ACS
```

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

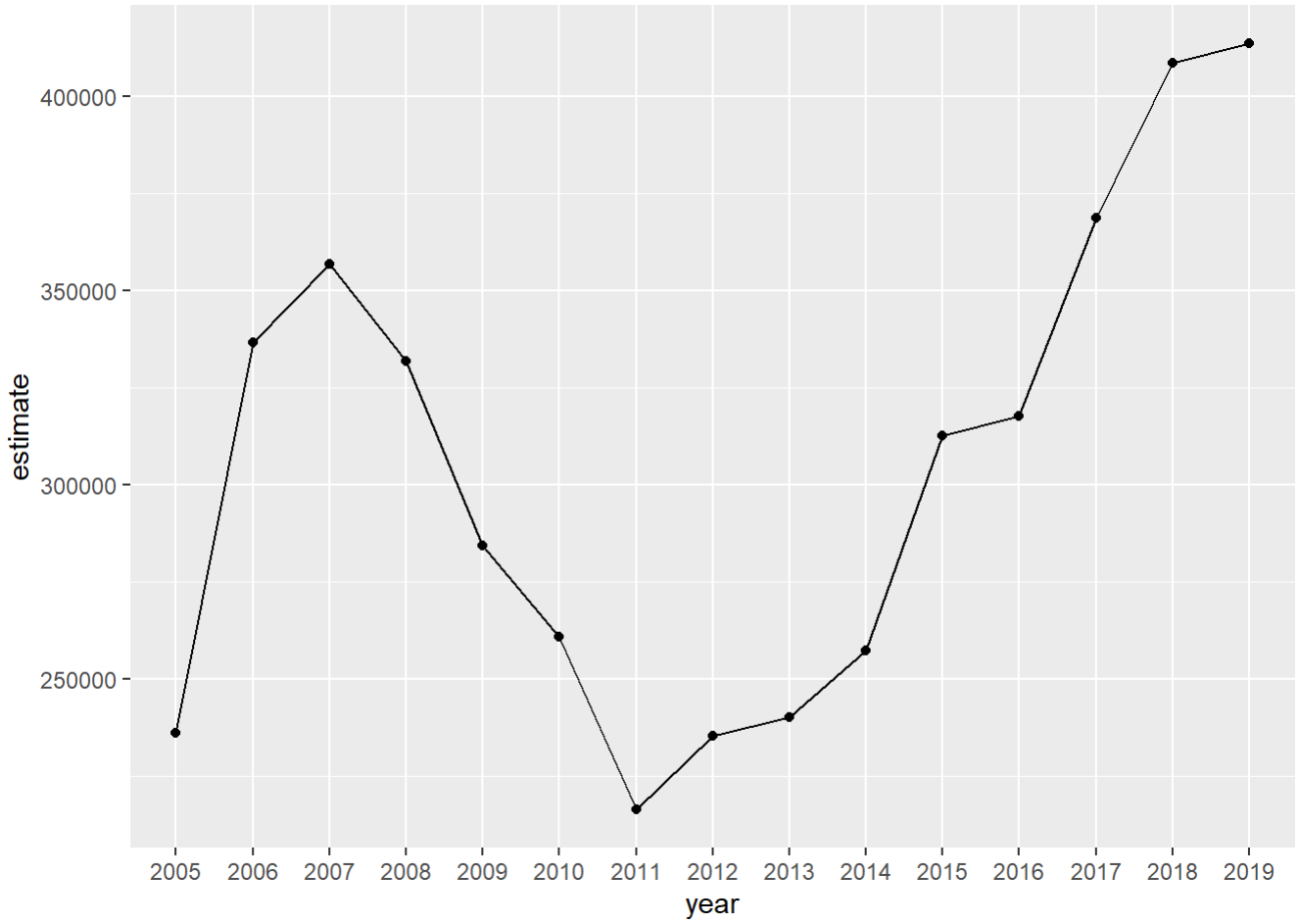
```
## Getting data from the 2018 1-year ACS
```

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

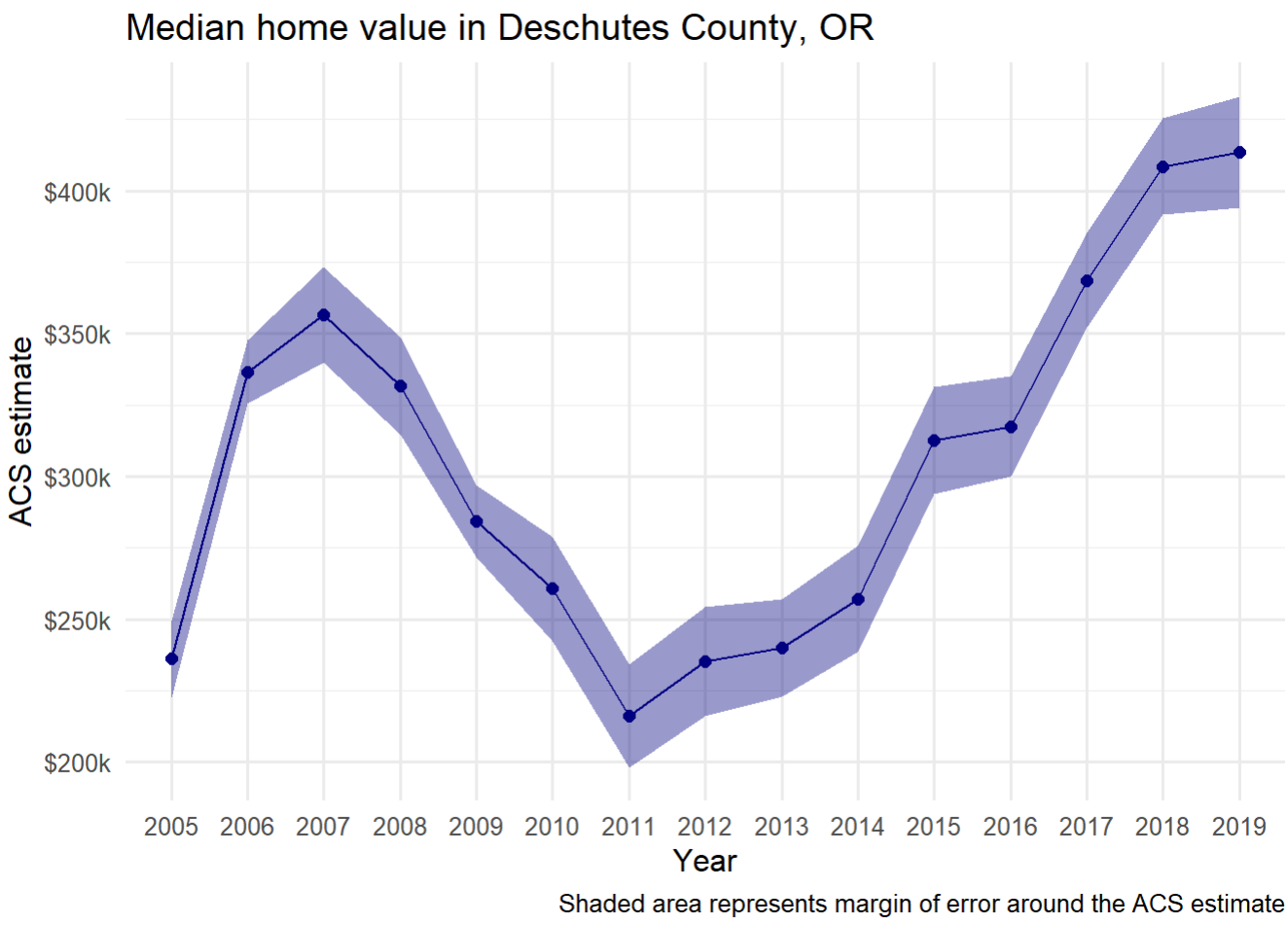
```
## Getting data from the 2019 1-year ACS
```

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

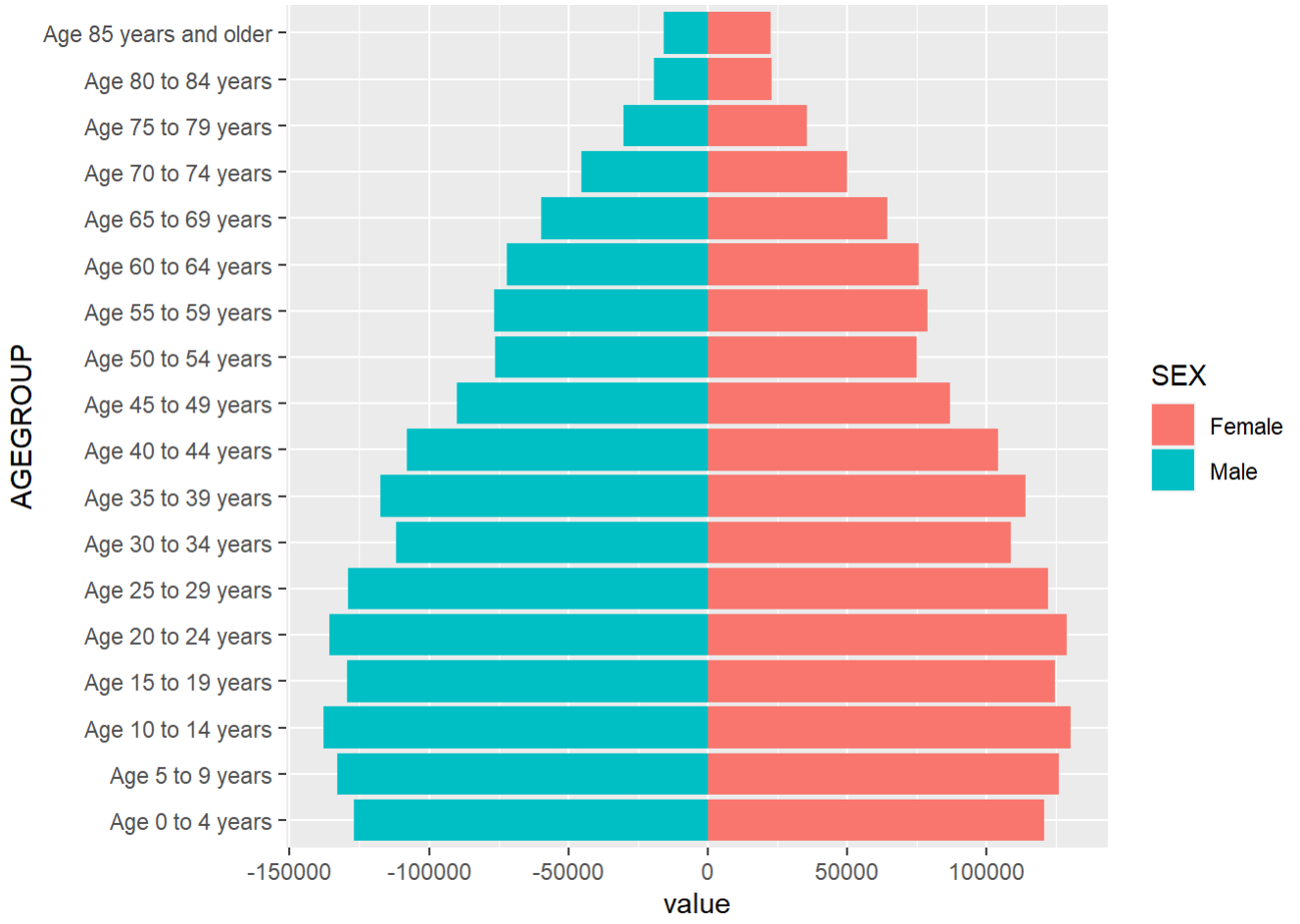
```
ggplot(deschutes_value, aes(x = year, y = estimate, group = 1)) +
  geom_line() +
  geom_point()
```




```
ggplot(deschutes_value, aes(x = year, y = estimate, group = 1)) +
  geom_ribbon(aes(ymax = estimate + moe, ymin = estimate - moe),
              fill = "navy",
              alpha = 0.4) +
  geom_line(color = "navy") +
  geom_point(color = "navy", size = 2) +
  theme_minimal(base_size = 12) +
  scale_y_continuous(labels = label_dollar(scale = .001, suffix = "k")) +
  labs(title = "Median home value in Deschutes County, OR",
       x = "Year",
       y = "ACS estimate",
       caption = "Shaded area represents margin of error around the ACS estimate")
```




```
utah <- get_estimates(
  geography = "state",
  state = "UT",
  product = "characteristics",
  breakdown = c("SEX", "AGEGROUP"),
  breakdown_labels = TRUE,
  year = 2019
)

utah_filtered <- filter(utah, str_detect(AGEGROUP, "^Age"),
                        SEX != "Both sexes") %>%
  mutate(value = ifelse(SEX == "Male", -value, value))

ggplot(utah_filtered, aes(x = value, y = AGEGROUP, fill = SEX)) +
  geom_col()
```

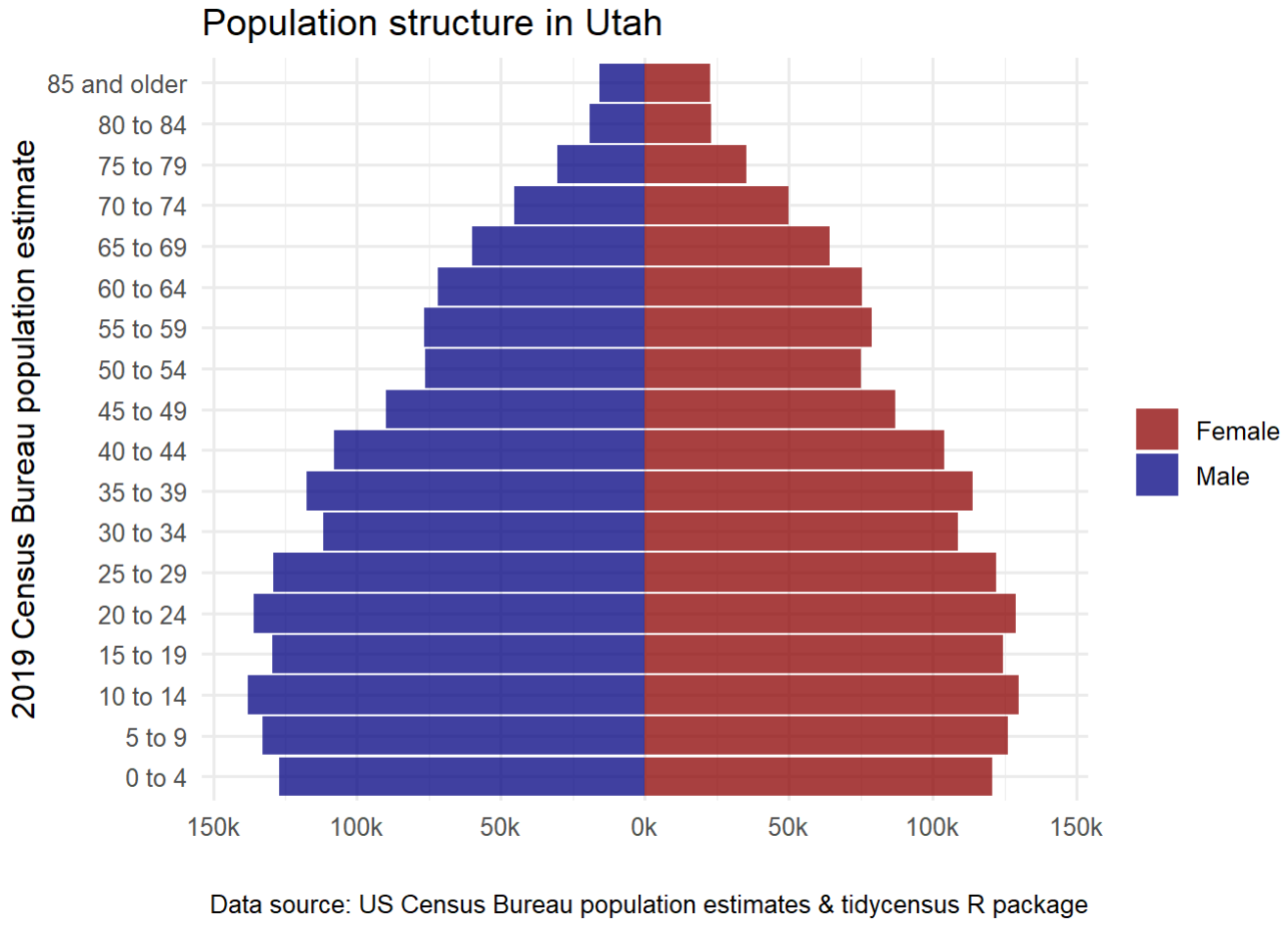



```
utah_pyramid <- ggplot(utah_filtered,
                       aes(x = value,
                           y = AGEGROUP,
                           fill = SEX)) +
  geom_col(width = 0.95, alpha = 0.75) +
  theme_minimal(base_family = "Verdana",
                base_size = 12) +
  scale_x_continuous(
    labels = ~ number_format(scale = .001, suffix = "k")(abs(.x)),
    limits = 140000 * c(-1,1)
  ) +
  scale_y_discrete(labels = ~ str_remove_all(.x, "Age\\s|\\syears")) +
  scale_fill_manual(values = c("darkred", "navy")) +
  labs(x = "",
       y = "2019 Census Bureau population estimate",
       title = "Population structure in Utah",
       fill = "",
       caption = "Data source: US Census Bureau population estimates & tidycensus R package")

utah_pyramid
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```




```
housing_val <- get_acs(
  geography = "tract",
  variables = "B25077_001",
  state = "OR",
  county = c(
    "Multnomah",
    "Clackamas",
    "Washington",
    "Yamhill",
    "Marion",
    "Columbia"
  ),
  year = 2020
)
```

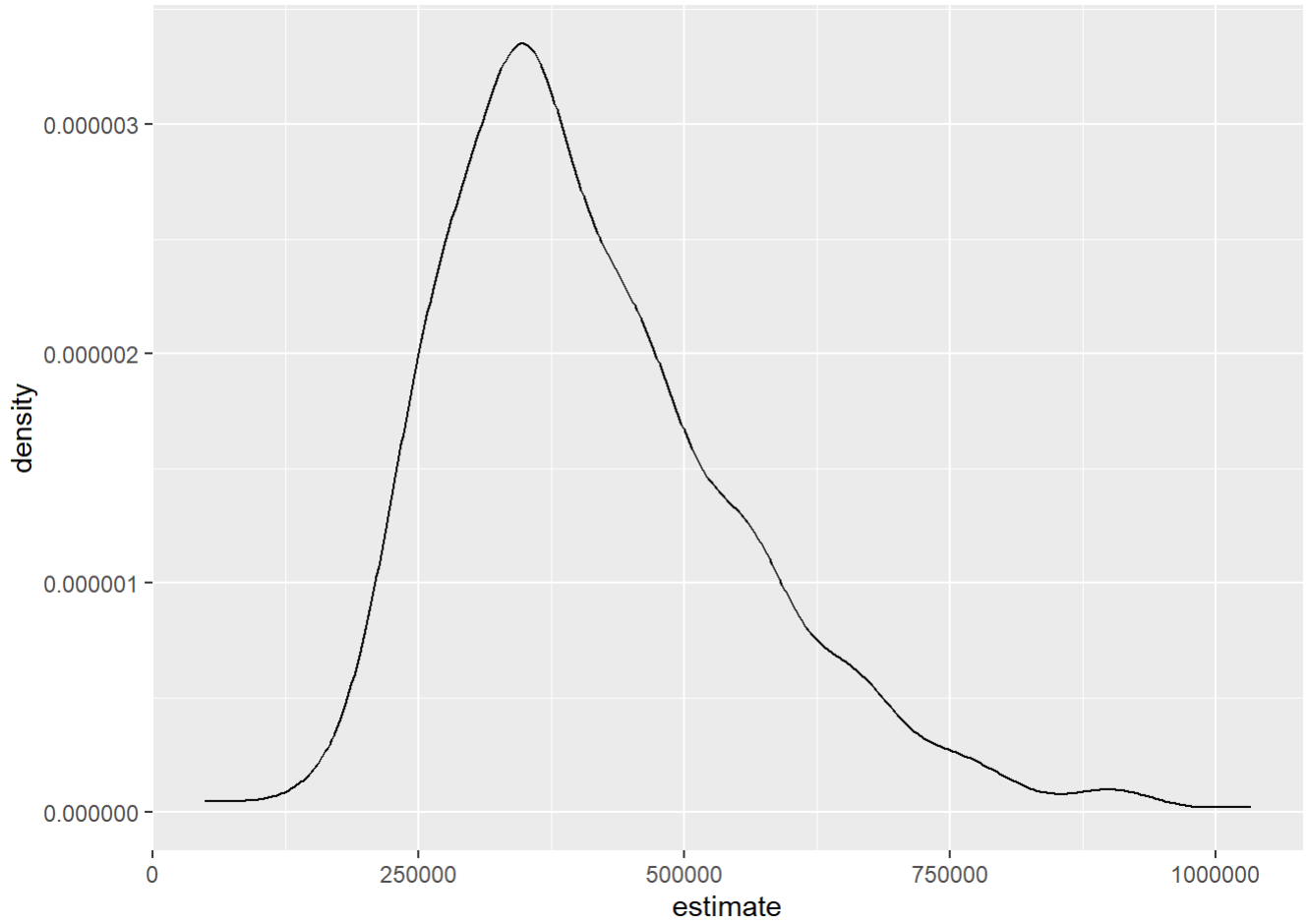
```
## Getting data from the 2016-2020 5-year ACS
```

```
housing_val2 <- separate(
  housing_val,
  NAME,
  into = c("tract", "county", "state"),
  sep = ", "
)

housing_val2 %>%
  group_by(county) %>%
  summarize(min = min(estimate, na.rm = TRUE),
            mean = mean(estimate, na.rm = TRUE),
            median = median(estimate, na.rm = TRUE),
            max = max(estimate, na.rm = TRUE))
```

```
## # A tibble: 6 × 5
##   county               min    mean median     max
##   <chr>              <dbl>   <dbl>  <dbl>   <dbl>
## 1 Clackamas County   62800 449941. 426700  909000
## 2 Columbia County   218100 277591. 275900  362200
## 3 Marion County      48700 270969. 261200  483500
## 4 Multnomah County  192900 455706. 425950 1033500
## 5 Washington County 221900 419618. 406100  769700
## 6 Yamhill County    230000 333816. 291100  545500
```

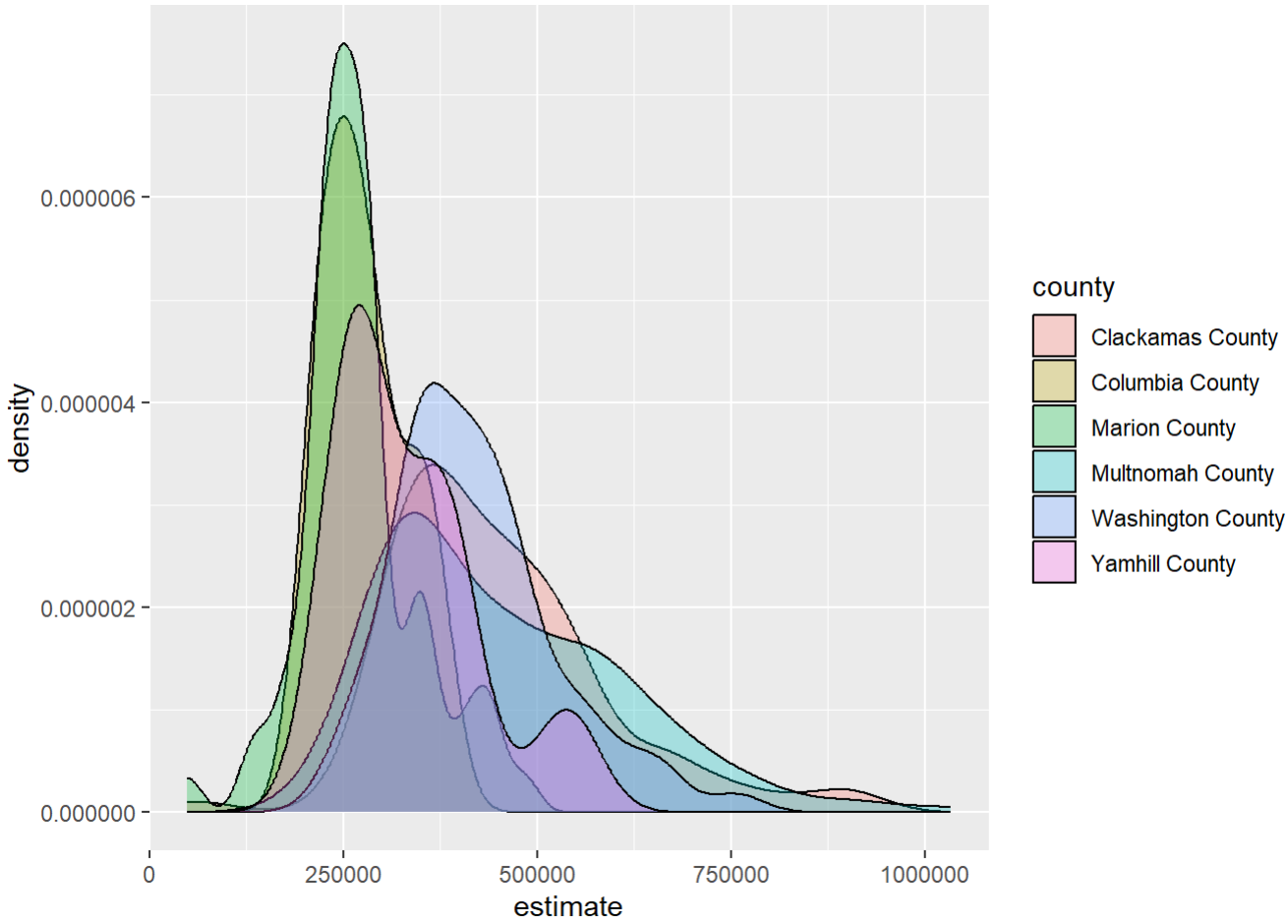
```
ggplot(housing_val2, aes(x = estimate)) +
  geom_density()
```

```
## Warning: Removed 9 rows containing non-finite values (`stat_density()`).
```



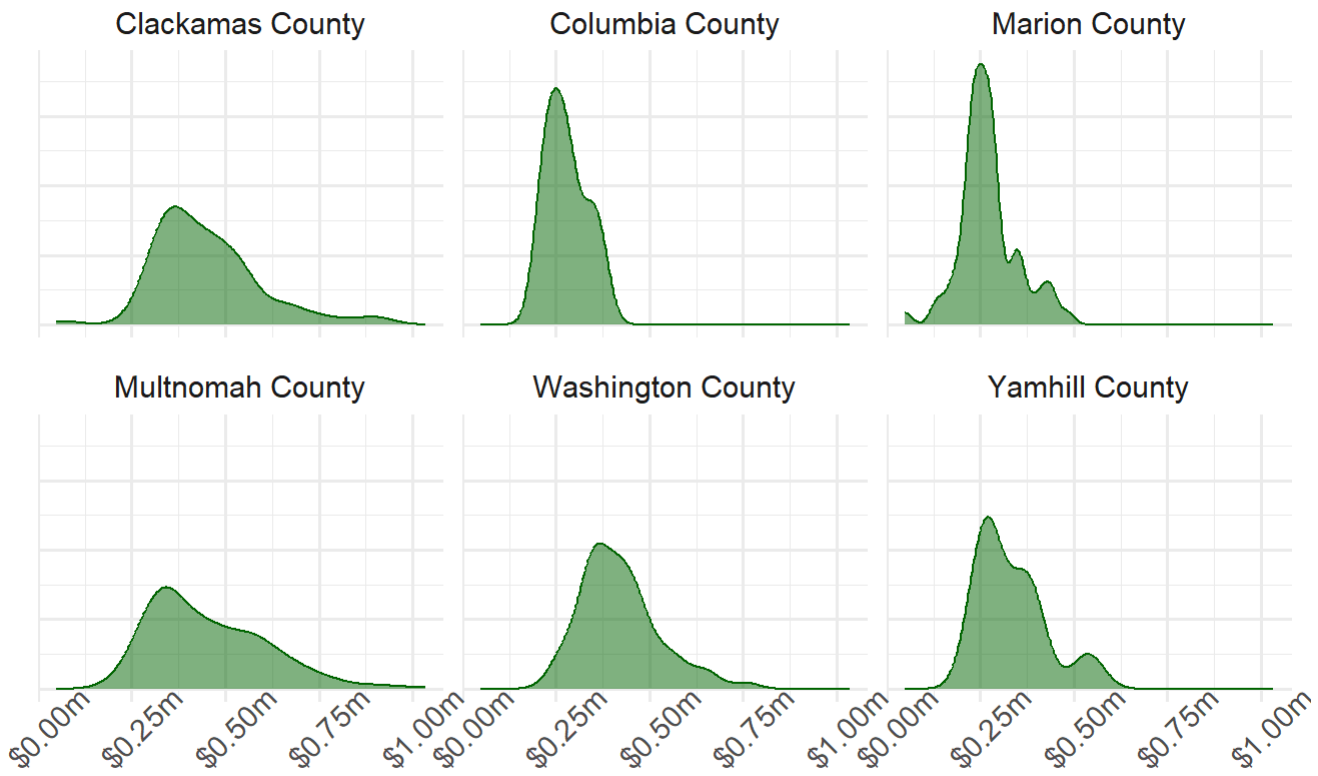

```
ggplot(housing_val2, aes(x = estimate, fill = county)) +
  geom_density(alpha = 0.3)
```

```
## Warning: Removed 9 rows containing non-finite values (`stat_density()`).
```




```
ggplot(housing_val2, aes(x = estimate)) +
  geom_density(fill = "darkgreen", color = "darkgreen", alpha = 0.5) +
  facet_wrap(~county) +
  scale_x_continuous(labels = dollar_format(scale = 0.000001,
                                            suffix = "m")) +
  theme_minimal(base_size = 14) +
  theme(axis.text.y = element_blank(),
        axis.text.x = element_text(angle = 45)) +
  labs(x = "ACS estimate",
       y = "",
       title = "Median home values by Census tract, 2015-2019 ACS")
```

```
## Warning: Removed 9 rows containing non-finite values (`stat_density()`).
```



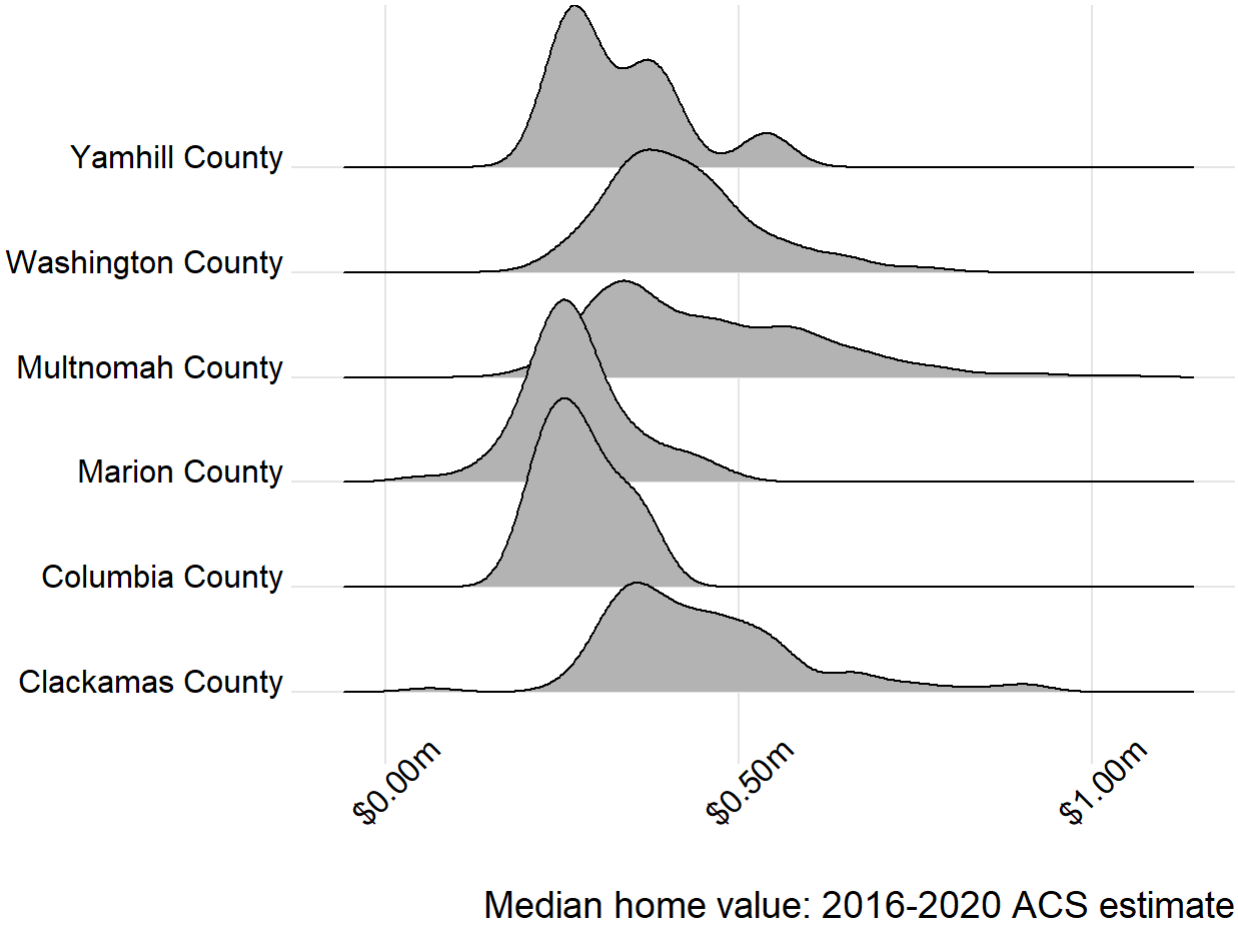

```
library(ggridges)
```

```
## Warning: package 'ggridges' was built under R version 4.2.2
```

```
ggplot(housing_val2, aes(x = estimate, y = county)) +
  geom_density_ridges() +
  theme_ridges() +
  labs(x = "Median home value: 2016-2020 ACS estimate",
       y = "") +
  scale_x_continuous(labels = label_dollar(scale = .000001, suffix = "m"),
                     breaks = c(0, 500000, 1000000)) +
  theme(axis.text.x = element_text(angle = 45))
```

```
## Picking joint bandwidth of 36200
```

```
## Warning: Removed 9 rows containing non-finite values (`stat_density_ridges()`).
```




```
library(ggbeeswarm)
```

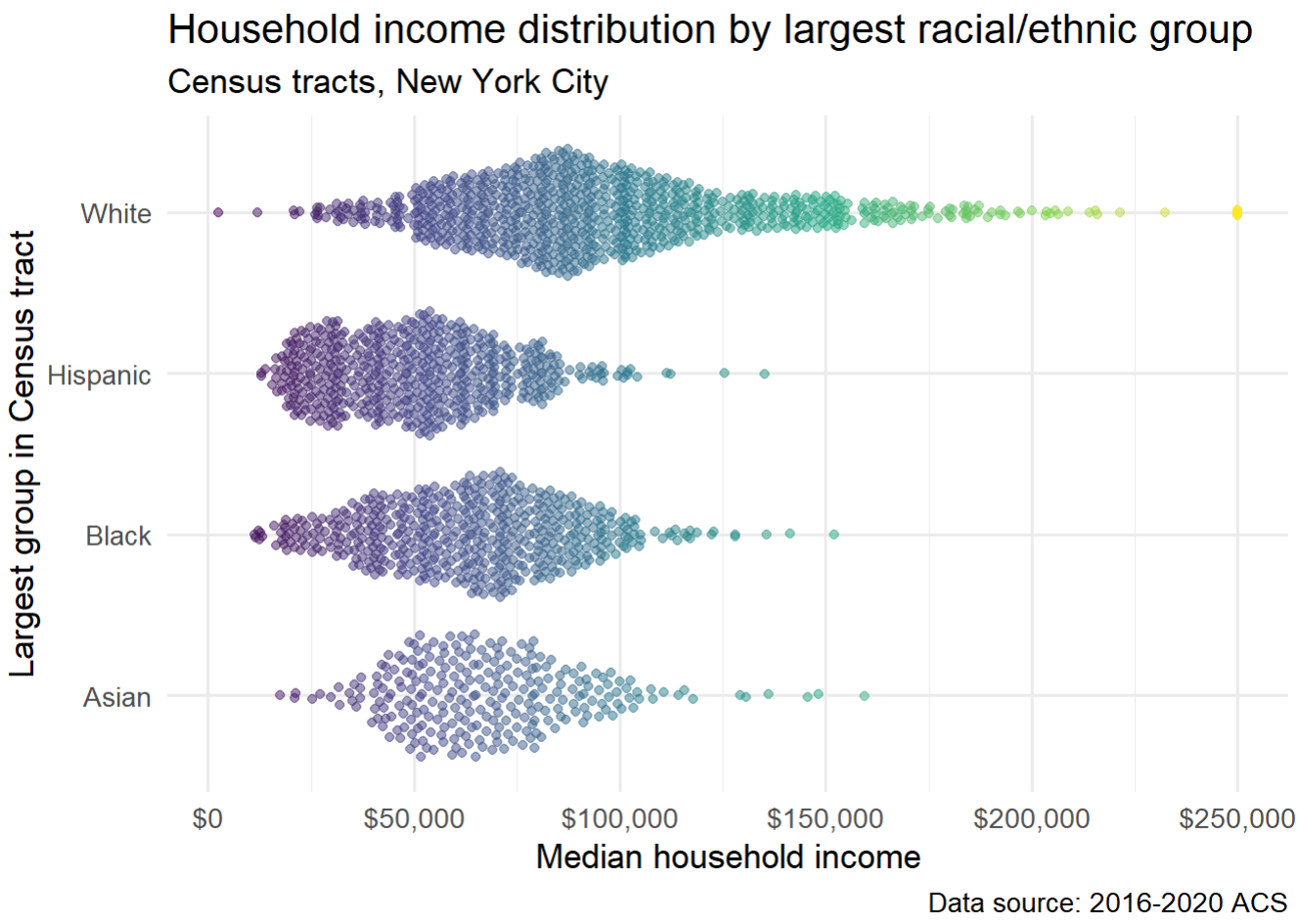
```
## Warning: package 'ggbeeswarm' was built under R version 4.2.2
```

```
ny_race_income <- get_acs(
  geography = "tract",
  state = "NY",
  county = c("New York", "Bronx", "Queens", "Richmond", "Kings"),
  variables = c(White = "B03002_003",
                Black = "B03002_004",
                Asian = "B03002_006",
                Hispanic = "B03002_012"),
  summary_var = "B19013_001",
  year = 2020
) %>%
  group_by(GEOID) %>%
  filter(estimate == max(estimate, na.rm = TRUE)) %>%
  ungroup() %>%
  filter(estimate != 0)
```

```
## Getting data from the 2016-2020 5-year ACS
```

```
ggplot(ny_race_income, aes(x = variable, y = summary_est, color = summary_est)) +
  geom_quasirandom(alpha = 0.5) +
  coord_flip() +
  theme_minimal(base_size = 13) +
  scale_color_viridis_c(guide = "none") +
  scale_y_continuous(labels = label_dollar()) +
  labs(x = "Largest group in Census tract",
       y = "Median household income",
       title = "Household income distribution by largest racial/ethnic group",
       subtitle = "Census tracts, New York City",
       caption = "Data source: 2016-2020 ACS")
```

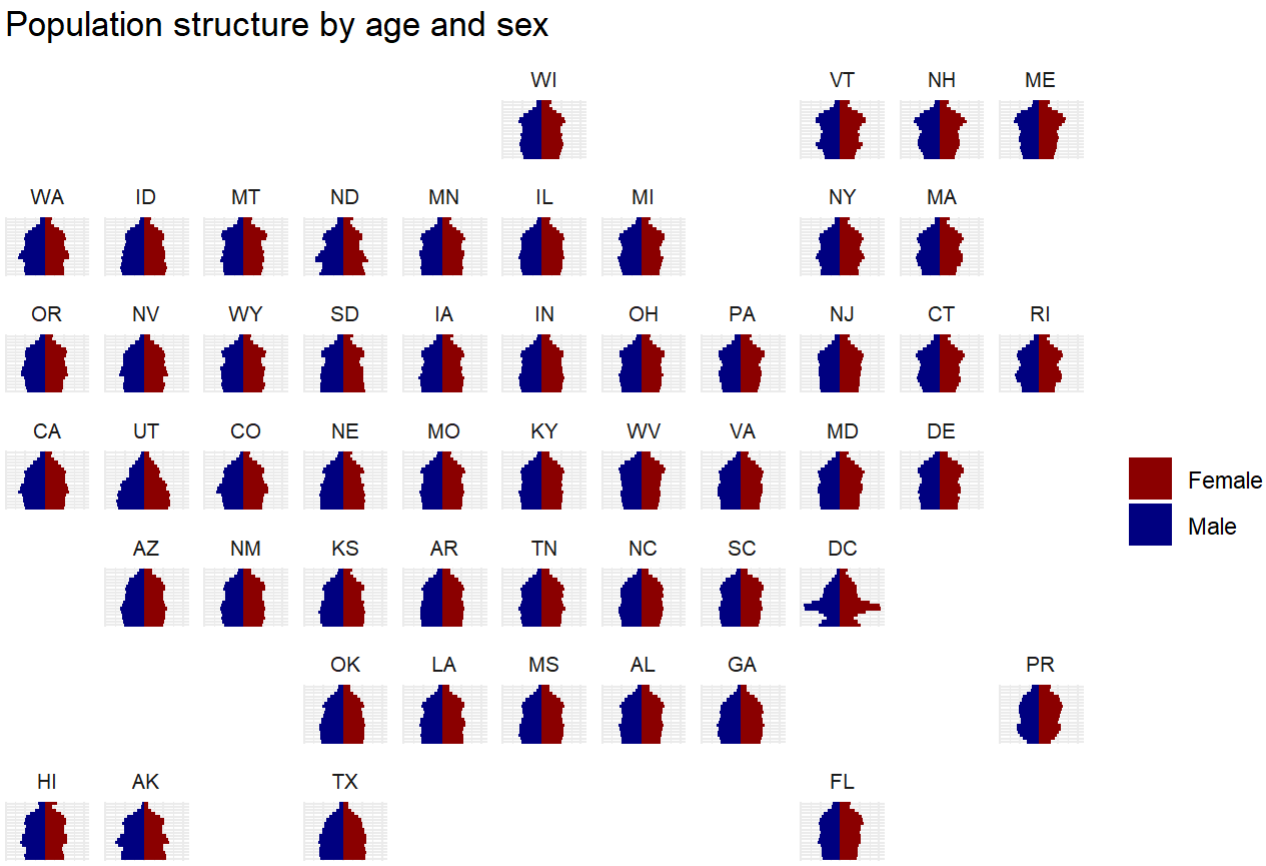
```
## Warning: Removed 35 rows containing missing values.
```




```
library(geofacet)
```

```
## Warning: package 'geofacet' was built under R version 4.2.2
```

```
us_pyramid_data <- get_estimates(
  geography = "state",
  product = "characteristics",
  breakdown = c("SEX", "AGEGROUP"),
  breakdown_labels = TRUE,
  year = 2019
) %>%
  filter(str_detect(AGEGROUP, "^Age"),
         SEX != "Both sexes") %>%
  group_by(NAME) %>%
  mutate(prop = value / sum(value, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(prop = ifelse(SEX == "Male", -prop, prop))

ggplot(us_pyramid_data, aes(x = prop, y = AGEGROUP, fill = SEX)) +
  geom_col(width = 1) +
  theme_minimal() +
  scale_fill_manual(values = c("darkred", "navy")) +
  facet_geo(~NAME, grid = "us_state_with_DC_PR_grid2",
            label = "code") +
  theme(axis.text = element_blank(),
        strip.text.x = element_text(size = 8)) +
  labs(x = "",
       y = "",
       title = "Population structure by age and sex",
       fill = "",
       caption = "Data source: US Census Bureau population estimates & tidycensus R package")
```




```
library(plotly)
```

```
## Warning: package 'plotly' was built under R version 4.2.2
```

```
##
## Attaching package: 'plotly'
```

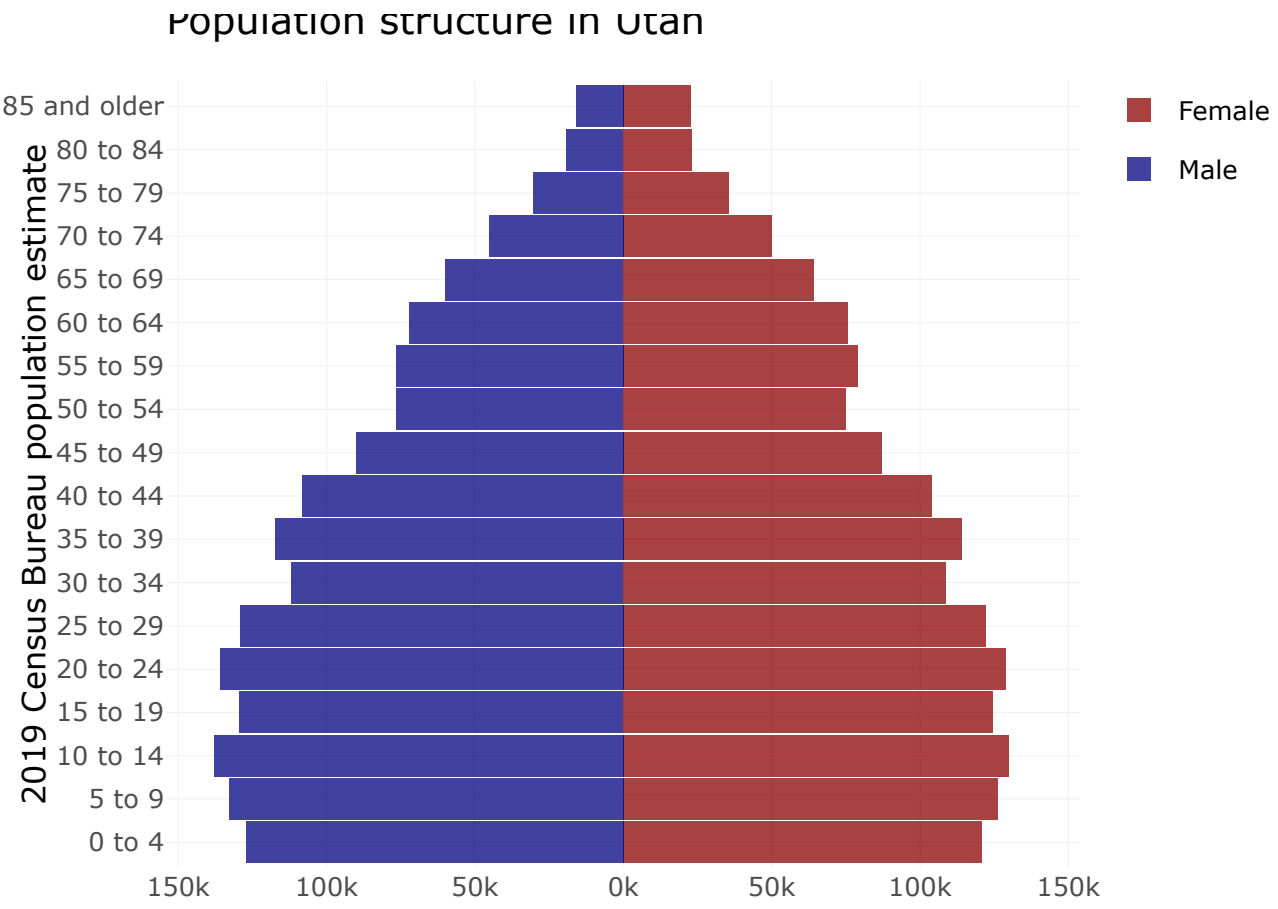
```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
ggplotly(utah_pyramid)
```

Population structure in Utah


Population structure in Utah
2019 Census Bureau population estimate
85 and older
80 to 84
75 to 79
70 to 74
65 to 69
60 to 64
55 to 59
50 to 54
45 to 49
40 to 44
35 to 39
30 to 34
25 to 29
20 to 24
15 to 19
10 to 14
5 to 9
0 to 4
150k
100k
50k
0k
50k
100k
150k
Female
Male