

Package ‘tidycensus’

May 19, 2021

Type Package

Title Load US Census Boundary and Attribute Data as 'tidyverse' and 'sf'-Ready Data Frames

Version 1.0

Date 2021-05-19

URL <https://github.com/walkerke/tidycensus>

BugReports <https://github.com/walkerke/tidycensus/issues>

Description

An integrated R interface to the decennial US Census and American Community Survey APIs and the US Census Bureau's geographic boundary files. Allows R users to return Census and ACS data as tidyverse-ready data frames, and optionally returns a list-column with feature geometry for all Census geographies.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.3.0)

Imports httr, sf, dplyr (>= 1.0.0), tigris, stringr, jsonlite (>= 1.5.0), purrr, rvest, tidyr (>= 1.0.0), rappdirs, readr, xml2, units, utils, rlang

Suggests ggplot2, survey, srvyr

RoxygenNote 7.1.1

NeedsCompilation no

Author Kyle Walker [aut, cre],
Matt Herman [aut],
Kris Eberwein [ctb]

Maintainer Kyle Walker <kyle@walker-data.com>

Repository CRAN

Date/Publication 2021-05-19 19:30:02 UTC

R topics documented:

| | |
|----------------|-----------|
| census_api_key | 2 |
| county_laea | 3 |
| fips_codes | 4 |
| get_acs | 5 |
| get_decennial | 7 |
| get_estimates | 9 |
| get_flows | 11 |
| get_pums | 13 |
| load_variables | 15 |
| mig_recodes | 16 |
| moe_product | 16 |
| moe_prop | 17 |
| moe_ratio | 17 |
| moe_sum | 18 |
| pums_variables | 18 |
| significance | 19 |
| state_laea | 20 |
| tidycensus | 20 |
| to_survey | 21 |
| Index | 22 |

| | |
|----------------|---|
| census_api_key | <i>Install a CENSUS API Key in Your .Renviron File for Repeated Use</i> |
|----------------|---|

Description

This function will add your CENSUS API key to your .Renviron file so it can be called securely without being stored in your code. After you have installed your key, it can be called any time by typing `Sys.getenv("CENSUS_API_KEY")` and can be used in package functions by simply typing `CENSUS_API_KEY`. If you do not have an .Renviron file, the function will create one for you. If you already have an .Renviron file, the function will append the key to your existing file, while making a backup of your original file for disaster recovery purposes.

Usage

```
census_api_key(key, overwrite = FALSE, install = FALSE)
```

Arguments

| | |
|-----------|---|
| key | The API key provided to you from the Census formatted in quotes. A key can be acquired at http://api.census.gov/data/key_signup.html |
| overwrite | If this is set to TRUE, it will overwrite an existing CENSUS_API_KEY that you already have in your .Renviron file. |
| install | if TRUE, will install the key in your .Renviron file for use in future sessions. Defaults to FALSE. |

Examples

```
## Not run:
census_api_key("111111abc", install = TRUE)
# First time, reload your environment so you can use the key without restarting R.
readRenviron("~/.Renviron")
# You can check it with:
Sys.getenv("CENSUS_API_KEY")

## End(Not run)

## Not run:
# If you need to overwrite an existing key:
census_api_key("111111abc", overwrite = TRUE, install = TRUE)
# First time, reload your environment so you can use the key without restarting R.
readRenviron("~/.Renviron")
# You can check it with:
Sys.getenv("CENSUS_API_KEY")

## End(Not run)
```

county_laea

County geometry with Alaska and Hawaii shifted and re-scaled

Description

Built-in dataset for use with `shift_geo = TRUE`

Dataset of US counties with Alaska and Hawaii shifted and re-scaled

Usage

```
data(county_laea)
```

```
data(county_laea)
```

Format

An object of class `sf` (inherits from `data.frame`) with 3143 rows and 2 columns.

Details

Dataset with county geometry for use when shifting Alaska and Hawaii

Built-in dataset for use with the `shift_geo` parameter, with the continental United States in a Lambert azimuthal equal area projection and Alaska and Hawaii counties and Census areas shifted and re-scaled. The data were originally obtained from the `albersusa` R package (<https://github.com/hrbrmstr/albersusa>).

`fips_codes`*Dataset with FIPS codes for US states and counties*

Description

Built-in dataset for smart state and county lookup. To access the data directly, issue the command `data(fips_codes)`.

- `county`: County name, title-case
- `county_code`: County code. (3-digit, 0-padded, character)
- `state`: Upper-case abbreviation of state
- `state_code`: State FIPS code (2-digit, 0-padded, character)
- `state_name`: Title-case name of state

Usage

```
data(fips_codes)
```

Format

An object of class `data.frame` with 3237 rows and 5 columns.

Details

Dataset with FIPS codes for US states and counties

Built-in dataset for use with the `lookup_code` function. To access the data directly, issue the command `data(fips_codes)`.

Note: this dataset includes FIPS codes for all counties that have appeared in the decennial Census or American Community Survey from 2010 to the present. This means that counties that have been renamed or absorbed into other geographic entities since 2010 remain in this dataset along with newly added or renamed counties.

If you need the FIPS codes and names for counties for a particular Census year, you can use the [counties](#) function from the `tigris` package and set the year parameter as required.

`get_acs`*Obtain data and feature geometry for the American Community Survey*

Description

Obtain data and feature geometry for the American Community Survey

Usage

```
get_acs(  
  geography,  
  variables = NULL,  
  table = NULL,  
  cache_table = FALSE,  
  year = 2019,  
  endyear = NULL,  
  output = "tidy",  
  state = NULL,  
  county = NULL,  
  zcta = NULL,  
  geometry = FALSE,  
  keep_geo_vars = FALSE,  
  shift_geo = FALSE,  
  summary_var = NULL,  
  key = NULL,  
  moe_level = 90,  
  survey = "acs5",  
  show_call = FALSE,  
  ...  
)
```

Arguments

| | |
|--------------------------|--|
| <code>geography</code> | The geography of your data. |
| <code>variables</code> | Character string or vector of character strings of variable IDs. tidycensus automatically returns the estimate and the margin of error associated with the variable. |
| <code>table</code> | The ACS table for which you would like to request all variables. Uses lookup tables to identify the variables; performs faster when variable table already exists through <code>load_variables(cache = TRUE)</code> . Only one table may be requested per call. |
| <code>cache_table</code> | Whether or not to cache table names for faster future access. Defaults to <code>FALSE</code> ; if <code>TRUE</code> , only needs to be called once per dataset. If variables dataset is already cached via the <code>load_variables</code> function, this can be bypassed. |

| | |
|---------------|--|
| year | The year, or endyear, of the ACS sample. 5-year ACS data is available from 2009 through 2019. 1-year ACS data is available from 2005 through 2019. Defaults to 2019. |
| endyear | Deprecated and will be removed in a future release. |
| output | One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns. |
| state | An optional vector of states for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to NULL. |
| county | The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. Defaults to NULL. |
| zcta | The zip code tabulation area(s) for which you are requesting data. Specify a single value or a vector of values to get data for more than one ZCTA. Numeric or character ZCTA GEOIDs are accepted. When specifying ZCTAs, geography must be set to "zcta" and 'state' and 'county' must be 'NULL'. Defaults to NULL. |
| geometry | if FALSE (the default), return a regular tibble of ACS data. if TRUE, uses the tigris package to return an sf tibble with simple feature geometry in the 'geometry' column. |
| keep_geo_vars | if TRUE, keeps all the variables from the Census shapefile obtained by tigris. Defaults to FALSE. |
| shift_geo | (deprecated) if TRUE, returns geometry with Alaska and Hawaii shifted for thematic mapping of the entire US. Geometry was originally obtained from the albersusa R package. As of May 2021, we recommend using <code>tigris::shift_geometry()</code> instead. |
| summary_var | Character string of a "summary variable" from the ACS to be included in your output. Usually a variable (e.g. total population) that you'll want to use as a denominator or comparison. |
| key | Your Census API key. Obtain one at https://api.census.gov/data/key_signup.html |
| moe_level | The confidence level of the returned margin of error. One of 90 (the default), 95, or 99. |
| survey | The ACS contains one-year, three-year, and five-year surveys expressed as "acs1", "acs3", and "acs5". The default selection is "acs5." |
| show_call | if TRUE, display call made to Census API. This can be very useful in debugging and determining if error messages returned are due to tidycensus or the Census API. Copy to the API call into a browser and see what is returned by the API directly. Defaults to FALSE. |
| ... | Other keyword arguments |

Value

A tibble or sf tibble of ACS data

Examples

```
## Not run:
library(tidycensus)
library(tidyverse)
library(viridis)
census_api_key("YOUR KEY GOES HERE")

tarr <- get_acs(geography = "tract", variables = "B19013_001",
               state = "TX", county = "Tarrant", geometry = TRUE)

ggplot(tarr, aes(fill = estimate, color = estimate)) +
  geom_sf() +
  coord_sf(crs = 26914) +
  scale_fill_viridis(option = "magma") +
  scale_color_viridis(option = "magma")

vt <- get_acs(geography = "county", variables = "B19013_001", state = "VT")

vt %>%
  mutate(NAME = gsub(" County, Vermont", "", NAME)) %>%
  ggplot(aes(x = estimate, y = reorder(NAME, estimate))) +
  geom_errorbarh(aes(xmin = estimate - moe, xmax = estimate + moe)) +
  geom_point(color = "red", size = 3) +
  labs(title = "Household income by county in Vermont",
       subtitle = "2015-2019 American Community Survey",
       y = "",
       x = "ACS estimate (bars represent margin of error)")

## End(Not run)
```

get_decennial

Obtain data and feature geometry for the decennial Census

Description

Obtain data and feature geometry for the decennial Census

Usage

```
get_decennial(
  geography,
  variables = NULL,
  table = NULL,
  cache_table = FALSE,
  year = 2010,
  sumfile = "sf1",
  state = NULL,
```

```

    county = NULL,
    geometry = FALSE,
    output = "tidy",
    keep_geo_vars = FALSE,
    shift_geo = FALSE,
    summary_var = NULL,
    key = NULL,
    show_call = FALSE,
    ...
)

```

Arguments

| | |
|---------------|--|
| geography | The geography of your data. |
| variables | Character string or vector of character strings of variable IDs. |
| table | The Census table for which you would like to request all variables. Uses lookup tables to identify the variables; performs faster when variable table already exists through <code>load_variables(cache = TRUE)</code> . Only one table may be requested per call. |
| cache_table | Whether or not to cache table names for faster future access. Defaults to <code>FALSE</code> ; if <code>TRUE</code> , only needs to be called once per dataset. If variables dataset is already cached via the <code>load_variables</code> function, this can be bypassed. |
| year | The year for which you are requesting data. 1990, 2000, and 2010 are available. |
| sumfile | The Census summary file. Defaults to <code>sf1</code> ; the function will look in <code>sf3</code> if it cannot find a variable in <code>sf1</code> . |
| state | The state for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to <code>NULL</code> . |
| county | The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. Defaults to <code>NULL</code> . |
| geometry | if <code>FALSE</code> (the default), return a regular tibble of ACS data. if <code>TRUE</code> , uses the <code>tigris</code> package to return an <code>sf</code> tibble with simple feature geometry in the 'geometry' column. state, county, tract, and block group are supported for 1990 through 2010; block and ZCTA geometry are supported for 2000 and 2010. |
| output | One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns. |
| keep_geo_vars | if <code>TRUE</code> , keeps all the variables from the Census shapefile obtained by <code>tigris</code> . Defaults to <code>FALSE</code> . |
| shift_geo | (deprecated) if <code>TRUE</code> , returns geometry with Alaska and Hawaii shifted for thematic mapping of the entire US. Geometry was originally obtained from the <code>albersusa</code> R package. As of May 2021, we recommend using <code>tigris::shift_geometry()</code> instead. |
| summary_var | Character string of a "summary variable" from the decennial Census to be included in your output. Usually a variable (e.g. total population) that you'll want to use as a denominator or comparison. |

| | |
|-----------|---|
| key | Your Census API key. Obtain one at https://api.census.gov/data/key_signup.html |
| show_call | if TRUE, display call made to Census API. This can be very useful in debugging and determining if error messages returned are due to tidycensus or the Census API. Copy to the API call into a browser and see what is returned by the API directly. Defaults to FALSE. |
| ... | Other keyword arguments |

Value

a tibble or sf tibble of decennial Census data

Examples

```
## Not run:
# Plot of race/ethnicity by county in Illinois for 2010
library(tidycensus)
library(tidyverse)
library(viridis)
census_api_key("YOUR KEY GOES HERE")
vars10 <- c("P005003", "P005004", "P005006", "P004003")

il <- get_decennial(geography = "county", variables = vars10, year = 2010,
                  summary_var = "P001001", state = "IL", geometry = TRUE) %>%
  mutate(pct = 100 * (value / summary_value))

ggplot(il, aes(fill = pct, color = pct)) +
  geom_sf() +
  facet_wrap(~variable)

## End(Not run)
```

get_estimates

Get data from the US Census Bureau Population Estimates APIs

Description

Get data from the US Census Bureau Population Estimates APIs

Usage

```
get_estimates(
  geography,
  product = NULL,
  variables = NULL,
  breakdown = NULL,
```

```

breakdown_labels = FALSE,
year = 2019,
state = NULL,
county = NULL,
time_series = FALSE,
output = "tidy",
geometry = FALSE,
keep_geo_vars = FALSE,
shift_geo = FALSE,
key = NULL,
show_call = FALSE,
...
)

```

Arguments

| | |
|------------------|--|
| geography | The geography of your data. |
| product | The data product (optional). "population", "components", "housing", and "characteristics" are supported. |
| variables | A character string or vector of character strings of requested variables to get from either the population, components, or housing API. |
| breakdown | The population breakdown used when product = "characteristics". Acceptable values are "AGEGROUP", "RACE", "SEX", and "HISP", for Hispanic/Not Hispanic. These values can be combined in a vector, returning population estimates in the value column for all combinations of these breakdowns. |
| breakdown_labels | Whether or not to label breakdown elements returned when product = "characteristics". Defaults to FALSE. |
| year | The data year (defaults to 2019) |
| state | The state for which you are requesting data. State names, postal codes, and FIPS codes are accepted. Defaults to NULL. |
| county | The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. Defaults to NULL. |
| time_series | If TRUE, the function will return a time series of observations back to the decennial Census of 2010. The returned column is either "DATE", representing a particular estimate date, or "PERIOD", representing a time period (e.g. births between 2016 and 2017), and contains integers representing those values. Integer to date or period mapping is available at https://www.census.gov/data/developers/data-sets/popest-popproj/popest/popest-vars/2019.html . |
| output | One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns. |
| geometry | if FALSE (the default), return a regular tibble of ACS data. if TRUE, uses the tigris package to return an sf tibble with simple feature geometry in the 'geometry' column. |

| | |
|---------------|--|
| keep_geo_vars | if TRUE, keeps all the variables from the Census shapefile obtained by tigris. Defaults to FALSE. |
| shift_geo | (deprecated) if TRUE, returns geometry with Alaska and Hawaii shifted for thematic mapping of the entire US. As of May 2021, we recommend using <code>tigris::shift_geometry()</code> instead. |
| key | Your Census API key. Obtain one at https://api.census.gov/data/key_signup.html . Can be stored in your <code>.Renviron</code> with <code>census_api_key("YOUR KEY", install = TRUE)</code> |
| show_call | if TRUE, display call made to Census API. This can be very useful in debugging and determining if error messages returned are due to tidy census or the Census API. Copy to the API call into a browser and see what is returned by the API directly. Defaults to FALSE. |
| ... | other keyword arguments |

Value

A tibble, or sf tibble, of population estimates data

| | |
|-----------|---|
| get_flows | <i>Obtain data and feature geometry for American Community Survey Migration Flows</i> |
|-----------|---|

Description

Obtain data and feature geometry for American Community Survey Migration Flows

Usage

```
get_flows(
  geography,
  variables = NULL,
  breakdown = NULL,
  breakdown_labels = FALSE,
  year = 2018,
  output = "tidy",
  state = NULL,
  county = NULL,
  msa = NULL,
  geometry = FALSE,
  key = NULL,
  moe_level = 90,
  show_call = FALSE
)
```

Arguments

| | |
|------------------|--|
| geography | The geography of your requested data. Possible values are "county", "county subdivision", and "metropolitan statistical area". MSA data is only available beginning with the 2009-2013 5-year ACS. |
| variables | Character string or vector of character strings of variable names. By default, 'get_flows()' returns the GEOID and names of the geographies as well as the number of people who moved in, out, and net movers of each geography ('MOVEDIN', 'MOVEDOUT', 'MOVEDNET'). If additional variables are specified, they are pulled in addition to the default variables. The names of additional variables can be found in the Census Migration Flows API documentation at https://api.census.gov/data/2018/acs/flows/variables.html . |
| breakdown | A character vector of the population breakdown characteristics to be crossed with migration flows data. For datasets between 2006-2010 and 2011-2015, selected demographic characteristics such as age, race, employment status, etc. are available. Possible values are "AGE", "SEX", "RACE", "HSGP", "REL", "HHT", "TEN", "ENG", "POB", "YEARS", "ESR", "OCC", "WKS", "SCHL", "AHINC", "APINC", and "HISP_ORIGIN". For more information and to see which characteristics are available in each year, visit the Census Migration Flows documentation at https://www.census.gov/data/developers/data-sets/acs-migration-flows.html . Note: not all characteristics are available in all years. |
| breakdown_labels | Whether or not to add columns with labels for the breakdown characteristic codes. Defaults to 'FALSE'. |
| year | The year, or endyear, of the ACS sample. The Migration Flows API is available for 5-year ACS samples from 2010 to 2018. Defaults to 2018. |
| output | One of "tidy" (the default) in which each row represents an enumeration unit-variable combination, or "wide" in which each row represents an enumeration unit and the variables are in the columns. |
| state | An optional vector of states for which you are requesting data. State names, postal codes, and FIPS codes are accepted. When requesting county subdivision data, you must specify at least one state. |
| county | The county for which you are requesting data. County names and FIPS codes are accepted. Must be combined with a value supplied to 'state'. |
| msa | The metropolitan statistical area for which you are requesting data. Specify a single value or a vector of values to get data for more than one MSA. Numeric or character MSA GEOIDs are accepted. When specifying MSAs, geography must be set to "metropolitan statistical area" and 'state' and 'county' must be 'NULL'. |
| geometry | if FALSE (the default), return a tibble of ACS Migration Flows data. If TRUE, return an sf object with the centroids of both origin and destination as 'sfc_POINT' columns. The origin point feature is returned in a column named 'centroid1' and is the active geometry column in the sf object. The destination point feature is returned in the 'centroid2' column. |
| key | Your Census API key. Obtain one at https://api.census.gov/data/key_signup.html |

| | |
|-----------|---|
| moe_level | The confidence level of the returned margin of error. One of 90 (the default), 95, or 99. |
| show_call | if TRUE, display call made to Census API. This can be very useful in debugging and determining if error messages returned are due to tidycensus or the Census API. Copy to the API call into a browser and see what is returned by the API directly. Defaults to FALSE. |

Value

A tibble or sf tibble of ACS Migration Flows data

Examples

```
## Not run:
get_flows(
  geography = "county",
  state = "VT",
  county = c("Washington", "Chittenden")
)

get_flows(
  geography = "county subdivision",
  breakdown = "RACE",
  breakdown_labels = TRUE,
  state = "NY",
  county = "Westchester",
  output = "wide",
  year = 2015
)

get_flows(
  geography = "metropolitan statistical area",
  variables = c("POP1YR", "POP1YRAGO"),
  geometry = TRUE,
  output = "wide",
  show_call = TRUE
)

## End(Not run)
```

get_pums

Load data from the American Community Survey Public Use Microdata Series API

Description

Load data from the American Community Survey Public Use Microdata Series API

Usage

```

get_pums(
  variables = NULL,
  state = NULL,
  puma = NULL,
  year = 2019,
  survey = "acs5",
  variables_filter = NULL,
  rep_weights = NULL,
  recode = FALSE,
  show_call = FALSE,
  key = NULL
)

```

Arguments

| | |
|------------------|--|
| variables | A vector of variables from the PUMS API. Use <code>View(pums_variables)</code> to browse variable options. |
| state | A state, or vector of states, for which you would like to request data. The entire US can be requested with <code>state = "all"</code> - though be patient with the data download! |
| puma | A vector of PUMAs from a single state, for which you would like to request data. To get data from PUMAs in more than one state, specify a named vector of state/PUMA pairs and set <code>state = "multiple"</code> . |
| year | The data year of the 1-year ACS sample or the endyear of the 5-year sample. Defaults to 2019. |
| survey | The ACS survey; one of either <code>"acs1"</code> or <code>"acs5"</code> (the default). |
| variables_filter | A named list of filters you'd like to return from the PUMS API. For example, passing <code>list(AGE = 25:50, SEX = 1)</code> will return only males aged 25 to 50 in your output dataset. Defaults to <code>NULL</code> , which returns all records. If a housing-only dataset is required, use <code>list(SPORDER = 1)</code> to only return householder records (taking care in your analysis to use the household weight <code>WGTP</code>). |
| rep_weights | Whether or not to return housing unit, person, or both housing and person-level replicate weights for calculation of standard errors; one of <code>"person"</code> , <code>"housing"</code> , or <code>"both"</code> . |
| recode | If <code>TRUE</code> , recodes variable values using Census data dictionary and creates a new <code>*_label</code> column for each variable that is recoded. Available for 2017 - 2019 data. Defaults to <code>FALSE</code> . |
| show_call | If <code>TRUE</code> , display call made to Census API. This can be very useful in debugging and determining if error messages returned are due to <code>tidycensus</code> or the Census API. Copy to the API call into a browser and see what is returned by the API directly. Defaults to <code>FALSE</code> . |
| key | Your Census API key. Obtain one at https://api.census.gov/data/key_signup.html |

Value

A tibble of microdata from the ACS PUMS API.

Examples

```
## Not run:
get_pums(variables = "AGEP", state = "VT")
get_pums(variables = "AGEP", state = "multiple", puma = c("UT" = 35008, "NV" = 00403))
get_pums(variables = c("AGEP", "ANC1P"), state = "VT", recode = TRUE)
get_pums(variables = "AGEP", state = "VT", survey = "acs1", rep_weights = "person")

## End(Not run)
```

| | |
|----------------|---|
| load_variables | <i>Load variables from a decennial Census or American Community Survey dataset to search in R</i> |
|----------------|---|

Description

Load variables from a decennial Census or American Community Survey dataset to search in R

Usage

```
load_variables(year, dataset, cache = FALSE)
```

Arguments

| | |
|---------|--|
| year | The year for which you are requesting variables. Either the year or endyear of the decennial Census or ACS sample. 5-year ACS data is available from 2009 through 2018. 1-year ACS data is available from 2005 through 2019. |
| dataset | One of "sf1", "sf3", "acs1", "acs3", "acs5", "acs1/profile", "acs3/profile", "acs5/profile", "acs1/subject", "acs3/subject", or "acs5/subject". |
| cache | Whether you would like to cache the dataset for future access, or load the dataset from an existing cache. Defaults to FALSE. |

Value

A tibble of variables from the requested dataset.

Examples

```
## Not run:
v15 <- load_variables(2015, "acs5", cache = TRUE)
View(v15)

## End(Not run)
```

| | |
|-------------|--|
| mig_recodes | <i>Dataset with Migration Flows characteristic recodes</i> |
|-------------|--|

Description

Built-in dataset for Migration Flows code label lookup.

- `characteristic`: Characteristic variable name
- `code`: Characteristic value code
- `desc`: Characteristic value label
- `ordered`: Whether or not recoded value should be ordered factor

Usage

```
data(mig_recodes)
```

Format

An object of class `spec_tbl_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 120 rows and 4 columns.

Details

Dataset with Migration Flows characteristic recodes

Built-in dataset that is created from the [Migration Flows API documentation](#). This dataset contains labels for the coded values returned by the Census API and is used when `breakdown_labels = TRUE` in [get_flows](#).

| | |
|-------------|--|
| moe_product | <i>Calculate the margin of error for a derived product</i> |
|-------------|--|

Description

Calculate the margin of error for a derived product

Usage

```
moe_product(est1, est2, moe1, moe2)
```

Arguments

| | |
|-------------------|--|
| <code>est1</code> | The first factor in the multiplication equation (an estimate) |
| <code>est2</code> | The second factor in the multiplication equation (an estimate) |
| <code>moe1</code> | The margin of error of the first factor |
| <code>moe2</code> | The margin of error of the second factor |

Value

A margin of error for a derived product

| | |
|----------|---|
| moe_prop | <i>Calculate the margin of error for a derived proportion</i> |
|----------|---|

Description

Calculate the margin of error for a derived proportion

Usage

```
moe_prop(num, denom, moe_num, moe_denom)
```

Arguments

| | |
|-----------|--|
| num | The numerator involved in the proportion calculation (an estimate) |
| denom | The denominator involved in the proportion calculation (an estimate) |
| moe_num | The margin of error of the numerator |
| moe_denom | The margin of error of the denominator |

Value

A margin of error for a derived proportion

| | |
|-----------|--|
| moe_ratio | <i>Calculate the margin of error for a derived ratio</i> |
|-----------|--|

Description

Calculate the margin of error for a derived ratio

Usage

```
moe_ratio(num, denom, moe_num, moe_denom)
```

Arguments

| | |
|-----------|---|
| num | The numerator involved in the ratio calculation (an estimate) |
| denom | The denominator involved in the ratio calculation (an estimate) |
| moe_num | The margin of error of the numerator |
| moe_denom | The margin of error of the denominator |

Value

A margin of error for a derived ratio

| | |
|---------|--|
| moe_sum | <i>Calculate the margin of error for a derived sum</i> |
|---------|--|

Description

Generates a margin of error for a derived sum. The function requires a vector of margins of error involved in a sum calculation, and optionally a vector of estimates associated with the margins of error. If the associated estimates are not specified, the user risks inflating the derived margin of error in the event of multiple zero estimates. It is recommended to inspect your data for multiple zero estimates before using this function and setting the inputs accordingly.

Usage

```
moe_sum(moe, estimate = NULL, na.rm = FALSE)
```

Arguments

| | |
|----------|---|
| moe | A vector of margins of error involved in the sum calculation |
| estimate | A vector of estimates, the same length as moe, associated with the margins of error |
| na.rm | A logical value indicating whether missing values (including NaN) should be removed |

Value

A margin of error for a derived sum

See Also

https://www2.census.gov/programs-surveys/acs/tech_docs/accuracy/MultiyearACSAccuracyofData2015.pdf

| | |
|----------------|--|
| pums_variables | <i>Dataset with PUMS variables and codes</i> |
|----------------|--|

Description

Built-in dataset for variable name and code label lookup. To access the data directly, issue the command `data(pums_variables)`.

- survey: acs1 or acs5
- year: Year of data. For 5-year data, last year in range.
- var_code: Variable name
- var_label: Variable label

- `data_type`: chr or num
- `level`: housing or person
- `val_min`: For numeric variables, the minimum value
- `val_max`: For numeric variables, the maximum value
- `val_label`: Value label
- `recode`: Use labels to recode values
- `val_length`: Length of value returned
- `val_na`: Value of NA value returned by API (if known)

Usage

```
data(pums_variables)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 31759 rows and 12 columns.

Details

Dataset with PUMS variables and codes

Built-in dataset that is created from the [Census PUMS data dictionaries](#). Use this dataset to lookup the names of variables to use in [get_pums](#). This dataset also contains labels for the coded values returned by the Census API and is used when `recode = TRUE` in [get_pums](#).

Because variable names and codes change from year to year, you should filter this dataset for the survey and year of interest. NOTE: 2017 - 2019 (acs1 and acs5) variables are available.

| | |
|---------------------------|---|
| <code>significance</code> | <i>Evaluate whether the difference in two estimates is statistically significant.</i> |
|---------------------------|---|

Description

Evaluate whether the difference in two estimates is statistically significant.

Usage

```
significance(est1, est2, moe1, moe2, clevel = 0.9)
```

Arguments

| | |
|---------------------|---|
| <code>est1</code> | The first estimate. |
| <code>est2</code> | The second estimate |
| <code>moe1</code> | The margin of error of the first estimate |
| <code>moe2</code> | The margin of error of the second estimate |
| <code>clevel</code> | The confidence level. May by 0.9, 0.95, or 0.99 |

Value

TRUE if the difference is statistically significant, FALSE otherwise.

See Also

https://www.census.gov/content/dam/Census/library/publications/2018/acs/acs_general_handbook_2018_ch07.pdf

state_laea

State geometry with Alaska and Hawaii shifted and re-scaled

Description

Built-in dataset for use with `shift_geo = TRUE`

Dataset of US states with Alaska and Hawaii shifted and re-scaled

Usage

```
data(state_laea)
```

```
data(state_laea)
```

Format

An object of class `sf` (inherits from `data.frame`) with 51 rows and 2 columns.

Details

Dataset with state geometry for use when shifting Alaska and Hawaii

Built-in dataset for use with the `shift_geo` parameter, with the continental United States in a Lambert azimuthal equal area projection and Alaska and Hawaii shifted and re-scaled. The data were originally obtained from the `albersusa` R package (<https://github.com/hrbrmstr/albersusa>).

tidycensus

Return tidy data frames from the US Census Bureau API

Description

This package uses US Census Bureau data but is neither endorsed nor supported by the US Census Bureau.

Author(s)

Kyle Walker

| | |
|-----------|--|
| to_survey | <i>Convert a data frame returned by <code>get_pums()</code> to a survey object</i> |
|-----------|--|

Description

This helper function takes a data frame returned by `get_pums` and converts it to a `tbl_svy` from the `srvyr` `as_survey` package or a `svyrep.design` object from the `svrepdesign` package. You can then use functions from the `srvyr` or `survey` to calculate weighted estimates with replicate weights included to provide accurate standard errors.

Usage

```
to_survey(  
  df,  
  type = c("person", "housing"),  
  class = c("srvyr", "survey"),  
  design = "rep_weights"  
)
```

Arguments

| | |
|---------------------|---|
| <code>df</code> | A data frame with PUMS person or housing weight variables, most likely returned by <code>get_pums</code> . |
| <code>type</code> | Whether to use person or housing-level weights; either "housing" or "person" (the default). |
| <code>class</code> | Whether to convert to a <code>srvyr</code> or <code>survey</code> object; either "survey" or "srvyr" (the default). |
| <code>design</code> | The survey design to use when creating a survey object. Currently the only option is code"rep_weights"/. |

Value

A `tbl_svy` or `svyrep.design` object.

Examples

```
## Not run:  
pums <- get_pums(variables = "AGEP", state = "VT", rep_weights = "person")  
pums_design <- to_survey(pums, type = "person", class = "srvyr")  
survey::svymean(~AGEP, pums_design)  
  
## End(Not run)
```

Index

* datasets

- county_laea, 3
- fips_codes, 4
- mig_recodes, 16
- pums_variables, 18
- state_laea, 20

as_survey, 21

census_api_key, 2

counties, 4

county_laea, 3

fips_codes, 4

get_acs, 5

get_decennial, 7

get_estimates, 9

get_flows, 11, 16

get_pums, 13, 19, 21

load_variables, 15

mig_recodes, 16

moe_product, 16

moe_prop, 17

moe_ratio, 17

moe_sum, 18

pums_variables, 18

significance, 19

state_laea, 20

svrepdesign, 21

tidycensus, 20

to_survey, 21