Adding ACS variables

Amy Youngbloom

3/25/2022

ACS Variables in the ACMT

There are currently over 200 ACS variables built into the ACMT that can easily be accessed with the get_acmt_standard_array function. To view these variables, you can import the acsvars csv file, which is stored in the ACMT folder:

```
#Download ACSColumns document (this removes any prior edits you've made to the list) -- this cod
e is also built into the GeocoderACMT.R code
acs_columns_url <- "http://host.docker.internal:7000/ACSColumns.csv"
download.file(url = acs_columns_url, destfile = "ACMT/ACSColumns.csv")


acsvars<-readr::read_csv('ACMT/ACSColumns.csv')
head(acsvars)
```

```
## # A tibble: 6 x 6
##   acs_col var_name universe_col pretty_name_cou… pretty_name_pro…
##   <chr>   <chr>    <chr>        <chr>            <chr>
## 1 B23001… males_1… B23001_003   Total males age… Percent of tota…
## 2 B23001… males_2… B23001_010   Total males age… Percent of tota…
## 3 B23001… males_2… B23001_017   Total males age… Percent of tota…
## 4 B23001… males_2… B23001_024   Total males age… Percent of tota…
## 5 B23001… males_3… B23001_031   Total males age… Percent of tota…
## 6 B23001… males_3… B23001_038   Total males age… Percent of tota…
## # … with 1 more variable:
## #   acs_variable_name_to_interpolate_by_sum_boolean_mapping <lgl>
```

Adding ACS variables

In some cases, you may want to pull ACS variables that are not included in the ACSColumns.csv document. You can use the tidycensus function ‘load_variables()’ to view the available acs variables for a given year. Note that some variables do change from year to year, so pay attention to the year of the variable(s) you are pulling and verify that it is consistent for each year you are interested in.

```
acs.2013<-tidycensus::load_variables(year=2013, dataset='acs5', cache=TRUE)
head(acs.2013)
```

```
## # A tibble: 6 x 3
##   name       label                               concept
##   <chr>      <chr>                               <chr>
## 1 B00001_001 Estimate!!Total                     UNWEIGHTED SAMPLE COUNT OF THE POP…
## 2 B00002_001 Estimate!!Total                     UNWEIGHTED SAMPLE HOUSING UNITS
## 3 B01001_001 Estimate!!Total                     SEX BY AGE
## 4 B01001_002 Estimate!!Total!!Male               SEX BY AGE
## 5 B01001_003 Estimate!!Total!!Male!!Under 5… SEX BY AGE
## 6 B01001_004 Estimate!!Total!!Male!!5 to 9 … SEX BY AGE
```

Step 1: Create a dataset of variables to add

Once you have identified the variables that you would like to add to pull using the ACMT, you will need to add the variable information to the ACSColumns document by specifying the following: (1) the variable name (var_name), (2) the column code where the variable is located in the ACS (acs_col), (3) the code of the universe variable, if applicable, to use to calculated proportions (universe_col), (4) a clean name for counts (pretty_name_count), (5) a clean name for proportions (pretty_name_proportion)

To add variables, we first create a new data frame that includes the ACS variables you'd like to add along with the information listed above. Note that you need to also add any variables that will be used as the universe variable for variables that you are adding, if that universe variable is not already in the ACSColumns.csv dataset (i.e., 'Total population 1 year and older')

```
var_name<-c("same_house", "moved_same_county", "moved_same_state", "moved_different_state", "mov
ed_from_abroad", "total_pop_over_1year")
acs_col<-c("B07003_004", "B07003_007", "B07003_010", "B07003_013", "B07003_016", "B07003_001")
universe_col<-c("B07003_001", "B07003_001", "B07003_001", "B07003_001", "B07003_001", "") #be su
re to include a blank space ("") if there is no applicable universe variable

#clean names for the count and proportion
pretty_name_count<-c("Residents living in the same house as 1 year ago (count)", "Residents who
 moved within the same county (count)", "Residents who moved from a different county within the
 same state (count)", "Residents who moved from a different state (count)", "Residents who moved
from abroad (count)", "Total population 1 year and older (count)")
pretty_name_proportion<-c("Percent of residents living in the same house as 1 year ago (proporti
on of all residents 1 year and older)", "Percent of residents who moved within the same county
 (proportion of all residents 1 year and older)", "Percent of residents who moved from a differe
nt county within the same state (proportion of all residents 1 year and older)", "Percent of res
idents who moved from a different state (proportion of all residents 1 year and older)", "Percen
t of residents who moved from abroad (count)", "") #be sure to include a blank space ("") if the
re is no applicable universe variable

#add interpolation instructions to the dataset as well (also need to manually designate in step
 3)
acs_variable_name_to_interpolate_by_sum_boolean_mapping<-c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE) #
add in interpoloation instructions (TRUE = sum to interpolate)

acs_addvars<-data.frame(var_name, acs_col, universe_col, pretty_name_count, pretty_name_proporti
on, acs_variable_name_to_interpolate_by_sum_boolean_mapping)
```

#Step 2: Append new variables dataset to existing ACSColumns.csv file Next, we will bring in the ACSColumns.csv file, and append the data frame we have just created.

```
acsvariables<-read.csv("ACMT/ACSColumns.csv")
names(acsvariables)
```

```
## [1] "acs_col"
## [2] "var_name"
## [3] "universe_col"
## [4] "pretty_name_count"
## [5] "pretty_name_proportion"
## [6] "acs_variable_name_to_interpolate_by_sum_boolean_mapping"
```

```
##in case it imports with an index column, subset to relevant columns only:
acsvariables<-acsvariables %>%
  dplyr::select('acs_col', 'var_name', 'universe_col', 'pretty_name_count', 'pretty_name_proport
ion', 'acs_variable_name_to_interpolate_by_sum_boolean_mapping')

acsvariables_add<-rbind(acsvariables, acs_addvars)
```

Step 3: Designate interpolation specifications

You will also need to designate interpolation specifications in order to instruct the ACMT to use summation-based (TRUE) or average-based (FALSE) area-weighted interpolation for a given context measure (interpolation). Currently, this information is part of the GeocoderACMT.R code file. Navigate to this file, which is located in workspace folder, and find the chunk of code that looks like this:

```
acs_variable_name_to_interpolate_by_sum_boolean_mapping <- c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, ....
                                                            TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE)


names(acs_variable_name_to_interpolate_by_sum_boolean_mapping) <- c("B01001_001", "B01001_002",
"B01001_003", "B01001_004", "B01001_005", ....
"B23001_066", "B23001_073", "B23001_078", "B23001_083")
```

In the GeocoderACMT.R file, you will need to manually add the ACS variables that you are adding to the end of the list: *names(acs_variable_name_to_interpolate_by_sum_boolean_mapping)*

For the *acs_variable_name_to_interpolate_by_sum_boolean_mapping*, you will add a TRUE if data should be summed (i.e., for any variable that is a count), and a FALSE for any variable that should not be summed (i.e., Median income).

Once the GeocoderACMT.R is updated with that information, save and close the GeocoderACMT.R file.

Step 4: Read updated ACMT code

Next, you will read the code to reset the ACMT function with the added interpolation information.

```
source('GeocoderACMT.R')
```

Step 5: Write new ACSColumns.csv file with appended variables

Once the GeocoderACMT.R code has been read, we can write our updated ACSColumns.csv file. Note that when you run the code above to read the GeocoderACMT.R code, the ACSColumns.csv document is re-downloaded, so any prior change you made are removed. Thus, we wait to write the ACSColumns.csv document until *after* the GeocoderACMT.R file is updated and read.

```
#write your updated ACS variables file to overwrite the existing ACSColumns.csv document:
write.csv(acsvariables_add, "ACMT/ACSColumns.csv")
#read in the updated file and view the tail to ensure your new variables have been added:
acsvars<-read_csv('ACMT/ACSColumns.csv')
tail(acsvars)
```

```
## # A tibble: 6 x 7
##      X1 acs_col var_name universe_col pretty_name_cou… pretty_name_pro…
##   <dbl> <chr>   <chr>    <chr>        <chr>             <chr>
## 1   278 B07003… same_ho… B07003_001   Residents livin… Percent of resi…
## 2   279 B07003… moved_s… B07003_001   Residents who m… Percent of resi…
## 3   280 B07003… moved_s… B07003_001   Residents who m… Percent of resi…
## 4   281 B07003… moved_d… B07003_001   Residents who m… Percent of resi…
## 5   282 B07003… moved_f… B07003_001   Residents who m… Percent of resi…
## 6   283 B07003… total_p… <NA>         Total populatio… <NA>
## # … with 1 more variable:
## #   acs_variable_name_to_interpolate_by_sum_boolean_mapping <lgl>
```

Step 6: Use ACMT to pull newly added ACS variables

Now your variables are added and ready to be pulled.

```
head(new_acs_vars)
```

```
##                   names       values
## 1 same_house_proportion 6.425018e-01
## 2      same_house_count 3.976852e+04
```