Package 'ACSdownload'

April 28, 2023

Title Obtain American Community Survey Summary File Data Tables from Census Bureau FTP Site (for EJ analysis, etc.)

Description This R package helps you download and parse raw data from the Census Bureau American Community Survey 5-year Summary Files, providing demographic data at the block and tract levels of resolution. You can obtain data from the entire USA all at once using this package, for one or more tables. Typically the Census Bureau makes it easy to obtain data from one state at a time, not every block group in the US. There are roughly 220,000 block groups in the US, and around 74,000 tracts.
Some options for obtaining Census ACS data are listed here:
http://www.census.gov/programs-surveys/acs/data.html
Other data sources that may be relevant include Census geodatabases at http://www.census.gov/geo/maps-data/data/tiger-data.html and data at http://www.census.gov/geo/maps-data/data/gazetteer.html
For any imported/suggested packages not on CRAN, see http://ejanalysis.github.io .

Version 2.2.0

Date 2022-09-01

Suggests census2020download,
UScensus2010blocks,
analyze.stuff,
ejanalysis

URL https://github.com/ejanalysis/ACSdownload, http://ejanalysis.github.io, http://www.ejanalysis.com/

BugReports https://github.com/ejanalysis/ACSdownload/issues

Depends R (>= 3.1.0)

RoxygenNote 7.2.2

License MIT + file LICENSE

Repository GitHub

Author info@ejanalysis.com

Maintainer info@ejanalysis.com <info@ejanalysis.com>

NeedsCompilation no

LazyData true

Encoding UTF-8

Imports magrittr

R topics documented:

acsdefaultendyearhere . . . 3
acsdefaultendyearhere_func . . . 3
ACSdownload . . . 3
acsfirstyearavailablehere . . . 4
clean.mystates . . . 5
clean.sumlevel . . . 6
datafile . . . 7
download.datafiles . . . 7
download.geo . . . 8
download.lookup.acs . . . 9
format_est_moe . . . 11
format_for_acs_package . . . 12
geo . . . 13
geofile . . . 14
geoformat2018 . . . 14
geoformat2019 . . . 15
geoformat2020 . . . 16
get.acs . . . 17
get.acs.all . . . 23
get.bg . . . 24
get.datafile.prefix . . . 24
get.field.info . . . 25
get.lookup.acs . . . 26
get.lookup.file.name . . . 27
get.read.geo . . . 27
get.table.info . . . 29
get.table.info2 . . . 30
get.tracts . . . 31
get.url.prefix . . . 31
get.url.prefix.lookup.table . . . 32
get.zipfile.prefix . . . 32
getseqnumsviafilenames . . . 33
getstatesviafilenames . . . 33
gettablesviaseqnums . . . 34
join.geo.to.tablist . . . 34
list_files_ftp . . . 35
list_files_ftp_worker . . . 36
lookup.acs . . . 36
lookup.acs2018 . . . 38
lookup.acs2020 . . . 39
merge_tables . . . 40
nhgis . . . 40
nhgisfind . . . 42
nhgisfips . . . 43
nhgisread . . . 44
nhgisreadcodebook . . . 45
read.concat.states . . . 45
read.geo . . . 46
set.needed . . . 48
unzip.datafiles . . . 49

url.to.find.zipfile 50
which.seqfiles 51
zipfile 52

Index **53**

`acsdefaultendyearhere` *The default end.year of 5yr ACS datasets this package uses in functions*

Description

This is to make it easier to update the package over time. It should be updated as needed, along with acsfirstyearavailablehere

---

`acsdefaultendyearhere_func`
Earliest end.year available via this package

Description

Earliest end.year available via this package

Usage

```
acsdefaultendyearhere_func()
```

`ACSdownload` *Obtain American Community Survey Summary File Data Tables from Census Bureau FTP Site*

Description

THIS PACKAGE MAY BE MOSTLY OBSOLETE NOW THAT SOME CRAN PACKAGES OFFER MOST OF THESE TOOLS, ESPECIALLY SEE THE PACKAGE totalcensus (https://cran.r-project.org/web/packages/totalcensus/index.html) which helps download Census data from the Census FTP site. As of 10/2022, only the github (development) version has info for ACS 2016-2020. devtools::install_github("GL-Li/totalcensus")

This R package helps you download and parse (huge) raw data from the Census Bureau American Community Survey 5-year Summary Files, providing demographic data at the block and tract levels of resolution. You can obtain data from the entire USA all at once using this package, for one or more tables. Key function is `get.acs`, and also see `set.needed`, `nhgis`, and also see data(lookup.acs2020) HOWEVER, DATA FORMAT IS CHANGING FOR SUMMARY FILE ACS DATA: SEE `https://www.census.gov/programs-surveys/acs/data/summary-file/updates-to-acs-summary-Overview.html`

Details

Typically the Census Bureau makes it easy to obtain data from one state at a time, not every block group in the US. There are roughly 220,000 block groups in the US, and around 74,000 tracts. The key function in this package is `get.acs`

For ACS documentation, see http://www.census.gov/programs-surveys/acs.html
Several options for obtaining Census ACS data are now listed here:
http://www.census.gov/programs-surveys/acs/data.html
Limits on downloads via American Fact Finder (not all US tracts at once) are noted here: https://ask.census.gov/faq.php?id=5000&faqId=1653
Other data sources that may be relevant include Census geodatabases at
http://www.census.gov/geo/maps-data/data/tiger-data.html and data at
http://www.census.gov/geo/maps-data/data/gazetteer.html. Also see the help for `get.acs`

Author(s)

info@ejanalysis.com <info@ejanalysis.com>

References

http://ejanalysis.github.io
http://www.ejanalysis.com

See Also

tidycensus to use a key to request ACS or Decennial Census data, but is slow if you want all blockgroups in a state or nationwide. **proxistat** package for block group points (lat lon), or **acs** package (http://cran.r-project.org/web/packages/acs/index.html) which lets one obtain more limited amounts of ACS data but provides better tools for working with the data once obtained.

`acsfirstyearavailablehere`

The earliest end.year of 5yr ACS datasets this package handles

---

Description

This is to make it easier to update the package over time. It should be updated as needed, along with acsdefaultendyearhere

clean.mystates *Utility to Clean Names of States for get.acs*

Description

Utility function used by `get.acs` and `download.datafiles`

Usage

```
clean.mystates(mystates = "all", testing = FALSE)
```

Arguments

`mystates`	Character vector, optional. Defines which states, using 2-character abbreviations (case-insensitive), or 'all' for all available. Default is 'all' which is 50 states plus DC, BUT NO LONGER PR (& not AS, GU, MP, UM, VI, US). If mystates is not specified or is 'all' then this returns the default. If mystates is specified, returns those defaults that match any of mystates. If mystates includes DC, PR, or US, those are returned, but "AS" "GU" "MP" "UM" "VI" are removed. One element of the vector can be 50, 51, or 52, where 50 represents the 50 states, 51 = 50 plus DC, or 52 = the 51 plus PR. In other words, the default could be written as c(50,'DC','PR') or as c(51,'PR') or just 52. Redundant entries are dropped, e.g., c(51,'DC') becomes 51.
`testing`	Logical value, optional, FALSE by default. LIMITS STATES TO DC AND DE if TRUE.

Details

Not in FTP ACS Summary files and not returned by this function's default:
#53 60 AS American Samoa <NA>
#54 66 GU Guam <NA>
#55 69 MP Northern Mariana Islands <NA>
#58 78 VI U.S. Virgin Islands <NA>
#57 74 UM U.S. Minor Outlying Islands <NA>

Note: The function stops if any of mystates is not found in the full list that includes these, but it returns only those excluding these five above, so an invalid state like 'ZQ' causes an error but an invalid state like 'VI' is silently removed without any error.

Puerto Rico (PR, FIPS 72), the District of Columbia (DC, 11), and the 50 States are in ACS 2016-2020 and Census 2020. For info on Island Areas, see https://www.census.gov/programs-surveys/decennial-census/technical-documentation/island-areas-censuses.html The 2020 Island Areas Censuses (IAC) include data for American Samoa (AS, FIPS 60), Guam (GU, 66), the Commonwealth of the Northern Mariana Islands (MP, 69), and the U.S. Virgin Islands (VI. 78), but not U.S. Minor Outlying Islands (UM, 74). The 2020 Census counted people living in the U.S. Island Areas using a long-form questionnaire. Other surveys, such as the American Community Survey (ACS), are not conducted in the Island Areas. Therefore, the Census Bureau used a long-form questionnaire to meet the Island Areas' data needs for demographic, social, economic, and housing unit information. This long-form questionnaire was similar to the ACS questionnaire used in the 50 states, the District of Columbia, and Puerto Rico. The 2020 IAC Demographic Profile Summary File provides access to all of the data. It provides data down to the place, county subdivision, and

estate (USVI only) level - not block groups. The 2020 IAC Demographic Profile Summary File data are available through the Census Bureau's data exploration platform, data.census.gov. The 2020 IAC Demographic Profile Summary File is located on the U.S. Census Bureau's file transfer protocol (FTP) server at <https://www2.census.gov/programs-surveys/decennial/2020/data/island-areas/. The easiest way is to start at the 2020 Island Areas Censuses Data Products webpage at www.census.gov/programs-surveys/decennial-census/decade/2020/planning-management/release/2020-islandareas-data-products.html

Value

Returns character vector of 2-character State abbreviations, lower case.

See Also

`get.acs` and `download.datafiles` which use this, and `get.state.info` (from **ejanalysis** package) based on `lookup.states` or `data(lookup.states, package=proxistat)` using **proxistat** package

---

`clean.sumlevel` *Utility to Clean SUMLEVEL for get.acs*

Description

Utility function used by `get.acs`.

Usage

```
clean.sumlevel(sumlevel = "bg")
```

Arguments

`sumlevel`	Character vector (1+ elements), optional, 'bg' by default. See details above.

Details

Interprets as 'bg' any of these: 150, '150', 'blockgroup', 'block group', or 'bg' (or variants, ignoring case)
Interprets as 'tract' any of these: 140, '140', or 'tract' (or variants, ignoring case)
Interprets as 'both' any of these: 'both' or a vector that has both of the above terms (or variants, ignoring case).

Value

Returns 'both', 'tract', or 'bg' (or stops with error if cannot interpret sumlevel input)

See Also

`get.acs` which uses this

datafile *Get name(s) of data file(s) for ACS 5-year summary file data*

Description

Returns name(s) of data file(s) based on state(s), a sequence file number, and end year.

Usage

```
datafile(state.abbrev, seqfilenum, end.year = acsdefaultendyearhere_func())
```

Arguments

state.abbrev	Required vector of one or more 2-character state abbreviations like "DC"
seqfilenum	Required sequence file number(s) used by ACS 5-year summary file (can be a single value like "0022" or a vector)
end.year	Optional end year for 5-year summary file, as character or number

Value

Returns character element that is name of data file such as e20105de0017000 or m20105de0017000

See Also

get.acs

download.datafiles *Download American Community Survey 5-yr data files*

Description

Attempts to download data files (estimates and margins of error) for specified states and tables, from the US Census Bureau's FTP site for American Community Survey (ACS) 5-year summary file data.

Usage

```
download.datafiles(
  tables,
  end.year = acsdefaultendyearhere_func(),
  mystates = 52,
  folder = getwd(),
  testing = FALSE,
  attempts = 5,
  silent = FALSE
)
```

Arguments

tables	Required character vector of table numbers, such as c("B01001", "B03002")
end.year	Character element, optional, like "2012". Defines last of five years of summary file dataset.
mystates	Character vector, now optional - Default is 50 states + DC + PR here, but otherwise relies on clean.mystates
folder	Default is getwd()
testing	Default to FALSE. If TRUE, provides info on progress of download.
attempts	Default is 5, specifies how many tries (maximum) for unzipping before trying to redownload and then give up.
silent	Optional, default is FALSE. Whether progress info should be sent to standard output (like the screen).

Value

Effect is to download and save locally a number of data files.

See Also

get.distances which allows you to get distances between all points.

---

download.geo *Download GEO txt file(s) with geo information for ACS*

Description

Download text file from US Census Bureau with geographic information for American Community Survey. The geo file is used to join data file(s) to FIPS/GEOID/NAME/SUMLEVEL/CKEY. Used by get.acs

Usage

```
download.geo(
  mystates,
  end.year = acsdefaultendyearhere_func(),
  folder = getwd(),
  testing = FALSE,
  attempts = 5,
  silent = FALSE
)
```

Arguments

mystates	vector of character 2-letter State abbreviations specifying which are needed
end.year	Specifies end year of 5-year summary file such as '2020'
folder	folder to use for saving files - default is current working directory
testing	Default to FALSE. If TRUE, provides info on progress of download.
attempts	Default is 5, specifies how many tries (maximum) for unzipping before trying to redownload and then give up.
silent	Optional, default is FALSE. Whether progress info should be sent to standard output (like the screen)

Details

Downloads to the current working directory unless another folder is specified. In contrast to the data files, the geo file is not zipped so does not have to be unzipped once downloaded. Key functions used:

- `url.to.find.zipfile`
- `geofile`
- `data(lookup.states, package='proxistat')`

Value

Side effect is downloading the file.

See Also

`get.acs` which uses this, and `get.read.geo`

Examples

```
 ## Not run:
   download.geo("de")
   download.geo( c("pr", "dc") )

## End(Not run)
```

`download.lookup.acs` *Download File with Information about ACS 5-Year Summary File Tables*

Description

Download and read lookup table of information on American Community Survey (ACS) tables, from the Census Bureau, namely which sequence files on the FTP site contain which tables and which variables. NOTE: This is largely obsolete now that data(lookup.acs2013) and similar files for other years are in this package.

Usage

```
download.lookup.acs(
  end.year = acsdefaultendyearhere_func(),
  folder = getwd(),
  silent = FALSE
)
```

Arguments

`end.year`	Character, optional, like '2020', which specifies the 2016-2020 dataset. Defines which 5-year summary file to use, based on end-year. Can be acsfirstyearavailablehere or later. Data for end.year='2019' were released in December 2020, for example. The 2017-2021 American Community Survey 5-year estimates are scheduled to be released on Thursday, December 8, 2022.

folder	Optional path to where to download file to, defaults to current working directory.
silent	Optional, default is FALSE. Whether to send progress info to standard output (like the screen)

Details

The URL scheme for lookup tables or datasets varies by year, for example,
For 2012 it is ftp://ftp.census.gov/acs2012_5yr/summaryfile/Sequence_Number_and_Table_Number_Lookup.txt
and for 2014 is http://www2.census.gov/programs-surveys/acs/summary_file/2014/documentation/user_tools/ACS_5yr_Seq_Table_Number_Lookup.txt
and for 2016 is http://www2.census.gov/programs-surveys/acs/summary_file/2016/documentation/user_tools/ACS_5yr_Seq_Table_Number_Lookup.txt

The 2013-2017 dataset is here...

Data tables by state by seqfile were here:

https://www2.census.gov/programs-surveys/acs/summary_file/2017/data/5_year_seq_by_state/Alabama/Tracts_Block_Groups_Only/
such as
https://www2.census.gov/programs-surveys/acs/summary_file/2017/data/5_year_seq_by_state/Alabama/Tracts_Block_Groups_Only/20175al0001000.zip

Geographies in Excel spreadsheets were here:

https://www2.census.gov/programs-surveys/acs/summary_file/2017/documentation/geography/5yr_year_geo/
such as
https://www2.census.gov/programs-surveys/acs/summary_file/2017/documentation/geography/5yr_year_geo/ak.xlsx

Lookup table of sequence files and variables was here:

https://www2.census.gov/programs-surveys/acs/summary_file/2017/documentation/user_tools/ACS_5yr_Seq_Table_Number_Lookup.txt

The 2014-2018 folders are here:
https://www2.census.gov/programs-surveys/acs/summary_file/2018/documentation/geography/
https://www2.census.gov/programs-surveys/acs/summary_file/2018/documentation/user_tools/

Value

By default, returns a data.frame with these fields:

- $ Table.ID : chr "B00001" "B00001" "B00001" "B00002" ...
- $ Sequence.Number : chr "0001" "0001" "0001" "0001" ...
- $ Line.Number : num NA NA 1 NA NA 1 NA NA 1 2 ...

- $ Start.Position : num 7 NA NA 8 NA NA 7 NA NA NA ...
- $ Total.Cells.in.Table : chr "1 CELL" "" "" "1 CELL" ...
- $ Total.Cells.in.Sequence: num NA NA NA 2 NA NA NA NA NA NA ...
- $ Table.Title : chr "UNWEIGHTED SAMPLE COUNT OF THE POPULATION" "Universe: Total population" "Total" "UNWEIGHTED SAMPLE HOUSING UNITS" ...
- $ Subject.Area : chr "Unweighted Count" "" "" "Unweighted Count" ...

For ACS 2008-2012:
length(my.lookup[,1])
[1] 24741
names(my.lookup)
[1] "File.ID" "Table.ID" "Sequence.Number" "Line.Number" "Start.Position"
[6] "Total.Cells.in.Table" "Total.Cells.in.Sequence" "Table.Title" "Subject.Area"

See Also

acs.lookup which does something similar but is more flexible & robust. Also see get.lookup.acs which does the same without downloading file – uses the copy in data() Also see data(lookup.acs2013) and similar data for other years. Also see get.acs, get.lookup.file.name, get.url.prefix.lookup.table

Examples

```
## Not run:
lookup.acs <- download.lookup.acs()

## End(Not run)
```

`format_est_moe` *Reorder cols of estimates and MOE table*

Description

Start with a table that has all the estimates columns together, followed by all the MOEs columns, and create a new column sort order so that estimates will be interspersed with (next to) their MOE values, as FactFinder format provides.

Usage

```
format_est_moe(my.list.of.tables)
```

Arguments

my.list.of.tables	Required list of tables from earlier steps in get.acs

Value

List of tables like input but with columns sorted in a new order.

See Also

get.acs and intersperse

format_for_acs_package *Reformat ACS Data Obtained by* get.acs *for Import to the acs Package*

Description

Work in progress *****

Currently only works for 5-year summary file data from ACS. Use same format as American Fact Finder uses for downloaded csv of tract data, for example. Format is ESTIMATE, MOE, ESTIMATE, MOE... and KEY cols are GEOID, FIPS, AND NAME, but also SUMLEVEL (specifies if tract or blockgroup, for example), and not STUSAB (2-letter State abbreviation).

Usage

```
format_for_acs_package(
  x,
  tableid = "",
  folder = getwd(),
  end.year = acsdefaultendyearhere_func(),
  savefile = TRUE
)
```

Arguments

x	Required list of tables from earlier steps in get.acs
tableid	Used to name any saved file. Should be a string such as 'B01001'. Default is ''
folder	Default is getwd() and specifies where to save csv if savefile=TRUE
end.year	Optional, text to use in filename if savefile=TRUE. The acs package needs this in the filename to infer the year, or that can be specified as the endyear parameter in read.acs
savefile	Default is TRUE which means save a csv file to folder

Details

Downloading C17002 from American Fact Finder results in a zip file with csv as follows:
ACS_13_5YR_B01001_with_ann.csv or ACS_12_5YR_C17002_with_ann.csv is format of the data file with estimates and MOE values
First row is header with field names. Other rows are tract data.
Note that the fields are:
GEOID, FIPS, NAME, e1, m1, e2, m2, etc.
Columns 1,2,3 are geo information. Columns 4+ are data (estimate,moe,estimate,moe, etc.)
First 2 rows example:
GEO.id,GEO.id2,GEO.display-label,HD01_VD01,HD02_VD01,HD01_VD02,HD02_VD02,HD01_VD03,HD02_VD0
0800000US11001500005000000501,11001500005000000501,"Census Tract 5.01, Washington city, Washington city, District of Columbia, District of Columbia",3113,296,232,164,50,47,77,84,82,90,199,122,19,29,24

ACS_12_5YR_C17002_metadata.csv has the long and short variable names.
There is no header row. A header of field names would be these 2: "short.name", "long.name"

Each row here corresponds to one column of the data/moe fields (after the geo fields) in the main data file.
First few rows example:
GEO.id,Id This is like GEOID field in acs via ftp GEO.id2,Id2 This seems to be like FIPS string portion of GEOID
GEO.display-label,Geography This is a full place name (NAME)
HD01_VD01,Estimate; Total:
HD02_VD01,Margin of Error; Total:
HD01_VD02,Estimate; Total: - Under .50
HD02_VD02,Margin of Error; Total: - Under .50

e.g.,
GEO.id GEO.id2 GEO.display.label HD01_VD01
1 Id Id2 Geography Estimate; Total:
2 1400000US24031700101 24031700101 Census Tract 7001.01, Montgomery County, Maryland 4477
3 1400000US24031700103 24031700103 Census Tract 7001.03, Montgomery County, Maryland 5776

JUST TRACTS were available from Fact Finder up to 2008-2012 ACS, so that is what acs package would typically import until recently.
Actually starting with 2009-2013 ACS, block groups are available via AFF, but only by specifying one (or each) county in a State.
##

Value

Data.frame for use in acs package.

See Also

`get.acs` to obtain acs data for use in this function, and then `read.acs` to read csv created by this function

---

`geo` *downloaded saved geographic information for 5-year summary file ACS dataset ending in given year*

Description

See `get.read.geo`. This data set is a geographic identifier file from the American Community Survey (ACS) 5-year summary file.

Format

A data.frame `'data.frame': e.g., maybe 294334 obs. of 5 variables: $ STUSAB : chr "AL" "AL" "AL" "AL" "AL" ... $ SUMLEVEL: chr "140" "140" "140" "140" ... $ GEOID : chr "14000US01001020100" "14000US01001020200" "14000US01001020300" "14000US01001020400" ... $ FIPS : chr "01001020100" "01001020200" "01001020300" "01001020400" ... $ KEY : chr "al0004656" "al0004657" "al0004658" "al0004659" ...`

Source

http://www2.census.gov/programs-surveys/acs

See Also

get.read.geo geofile download.geo

---

geofile *Get name(s) of GEO txt file(s) with geo information for ACS*

Description

Get name of text file used by US Census Bureau with geographic information for American Community Survey. That geo file can be used to join data file(s) to FIPS/GEOID/NAME/SUMLEVEL/CKEY.

Usage

```
geofile(mystates, end.year = acsdefaultendyearhere_func())
```

Arguments

mystates	vector of character 2-letter State abbreviations specifying which are needed
end.year	end.year of 5-year summary file such as '2020'

Value

Character vector of file names, example: "g20105md.txt" Note this is only needed once per state, not once per seqfile. (It might even be available as a single US file?)

See Also

get.acs and download.geo which uses this

---

geoformat2018 *geographic information for ACS dataset*

Description

This data set has the format used by geographic identifier files in the American Community Survey (ACS) 5-year summary file. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1, so 2014-2018 would be available by Dec 2019.

Format

A data.frame 'data.frame': In 2016 and 2017, for example, it had 53 obs. of 5 variables: $ varname : chr "FILEID" "STUSAB" "SUMLEVEL" "COMPONENT" ... $ description: chr "Always equal to ACS Summary File identification" "State Postal Abbreviation" "Summary Level" "Geographic Component" ... $ size : int 6 2 3 2 7 1 1 1 2 2 ... $ start : int 1 7 9 12 14 21 22 23 24 26 ... $ type : chr "Record" "Record" "Record" "Record" ...

Source

Table found in given year dataset, info at https://www.census.gov/programs-surveys/acs/technical-documentation.html

See Also

get.acs

---

geoformat2019 *geographic information for ACS dataset*

Description

This data set has the format used by geographic identifier files in the American Community Survey (ACS) 5-year summary file. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1, so 2015-2019 was available by Dec 2020.

Format

A data.frame

```
'data.frame': 53 obs. of  5 variables:
$ varname    : chr  "FILEID" "STUSAB" "SUMLEVEL" "COMPONENT" ...
$ description: chr  "Always equal to ACS Summary File identification" "State Postal Abbreviation" "S
$ size       : num  6 2 3 2 7 1 1 1 2 2 ...
$ start      : num  1 7 9 12 14 21 22 23 24 26 ...
$ type       : chr  "Record" "Record" "Record" "Record" ...
```

Source

Table found in given year dataset, info at https://www.census.gov/programs-surveys/acs/library/handbooks/geography.html https://www.census.gov/programs-surveys/acs/data/data-via-ftp.html https://www2.census.gov/programs-surveys/acs/summary_file/2020/data/5_year_entire_sf/2020_ACS_Geography_Files.zip

See Also

get.acs

geoformat2020 *geographic information for ACS dataset*

Description

This data set has the format used by geographic identifier files in the American Community Survey (ACS) 5-year summary file. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1, so 2016-2020 was available by Dec 2021.

Format

A data.frame

```
'data.frame': 53 obs. of  5 variables:  (at least for 2019 version)
$ varname    : chr  "FILEID" "STUSAB" "SUMLEVEL" "COMPONENT" ...
$ description: chr  "Always equal to ACS Summary File identification" "State Postal Abbreviation" "S
$ size       : num  6 2 3 2 7 1 1 1 2 2 ...
$ start      : num  1 7 9 12 14 21 22 23 24 26 ...
$ type       : chr  "Record" "Record" "Record" "Record" ...
```

Details

geoformat2020 <- geoformat2019 # if they are assumed to be the same. metadata <- list(ejscreen_releasedate = 'late 2022', ejscreen_version = '2.1', ACS_version = '2016-2020', ACS_releasedate = '3/17/2022') # attributes(geoformat2020) <- c(attributes(geoformat2020), metadata) # OR... try attr(x, which=names(metadata[n])) <- metadata[n] attr(geoformat2020, which = 'ejscreen_releasedate') <- 'late 2022' attr(geoformat2020, which = 'ejscreen_version') <- '2.1' attr(geoformat2020, which = 'ACS_version') <- '2016-2020' attr(geoformat2020, which = 'ACS_releasedate') <- '3/17/2022' usethis::use_data(geoformat2020)

see data at browseURL('https://www2.census.gov/programs-surveys/acs/summary_file/2019/data/5_year_seq_by_stat but that is not available in analogous URL for 2020.

##geo files: ## browseURL('https://www2.census.gov/programs-surveys/acs/summary_file/2019/documentation/geogra setwd('~/R/ACSdownload/inst/') download.file('https://www2.census.gov/programs-surveys/acs/summary_file/2019/do destfile = 'g20195us.csv') that is the full US geo file, not what I call the geoformat file (which is tiny)

2018 and 2019 lack the normal geo file documentation they used to have also, read.geo() used read.fwf() at least for older years. 2018 or 2019 might have csv not just txt format now?

there is this:

https://www2.census.gov/programs-surveys/acs/summary_file/2018/documentation/tech_docs/ACS_2018_SF_5YR_Ap

@source Table found in given year dataset, info at https://www.census.gov/programs-surveys/acs/library/handbooks/geography.html https://www.census.gov/programs-surveys/acs/data/data-via-ftp.html https://www2.census.gov/programs-surveys/acs/summary_file/2020/data/5_year_entire_sf/2020_ACS_Geography_Files.zip

See Also

get.acs

`get.acs` *Download Tables from American Community Survey (ACS) 5-year Summary File*

Description

NOTE: Census formats changed a lot, and some of the info below is obsolete. Also see newer get.acs.all() that will try to use new format for 5yr summary file ACS that makes it easier to get one table for every blockgroup in the US.

This function will download and parse 1 or more tables of data from the American Community Survey's 5-year Summary File FTP site, for all Census tracts and/or block groups in specified State(s). Estimates and margins of error are obtained, as well as long and short names for the variables, which can be specified if only parts of a table are needed.

It is especially useful if you want a lot of data such as all the blockgroups in the USA, which may take a long time to obtain using the Census API and a package like **tidycensus**

Release schedule for ACS 5-year data is here: https://www.census.gov/programs-surveys/acs/news/data-releases.html The 2016-2020 ACS 5-year estimates release date is December 9, 2021.

Usage

```
get.acs(
  tables = "B01001",
  mystates = "all",
  end.year = acsdefaultendyearhere_func(),
  base.path = getwd(),
  data.path = file.path(base.path, "acsdata"),
  output.path = file.path(base.path, "acsoutput"),
  sumlevel = "both",
  vars = "all",
  varsfile,
  new.geo = TRUE,
  write.files = FALSE,
  save.files = FALSE,
  write.acspkg = FALSE,
  testing = FALSE,
  noEditOnMac = FALSE,
  silent = FALSE,
  save.log = TRUE,
  filename.log = "log"
)
```

Arguments

`tables`	Character vector, optional. Defines tables to obtain, such as 'B01001' (the default). NOTE: If the user specifies a table called 'ejscreen' then a set of tables used by that tool are included. Those tables in EJScreen 2.0 are c("B01001", "B03002", "B15002", 'C16002', "C17002", "B25034", 'B23025')

`mystates` Character vector, optional, 'all' by default which means all available states plus DC and PR but not VI etc. Defines which States to include in downloads of data tables.

`end.year` Character, optional. Defines a valid ending year of a 5-year summary file. Can be '2020' for example. Not all years are tested. Actually works if numeric like 2017, instead of character, too.

`base.path` Character, optional, getwd() by default. Defines a base folder to start from, in case data.path and/or output.path are not specified, in which case a subfolder under base.path, called acsdata or acsoutput respectively, will be created and used for downloads or outputs of the function.

`data.path` Character, optional, file.path(base.path, 'acsdata') by default. Defines folder (created if does not exist) where downloaded files will be stored.

`output.path` Character, optional, file.path(base.path, 'acsoutput') by default. Defines folder (created if does not exist) where output files (results of this function) will be stored.

`sumlevel` Default is "both" (case insensitive) in which case tracts and block groups are returned. Also c('tract', 'bg') or c(140,150) and similar patterns work. If "tract" or 140 or some other match, but not block groups, is specified (insensitive to case, tract can be plural or part of a word, etc.), just tracts are returned. If "bg" or 150 or "blockgroups" or "block groups" or some other match (insensitive to case, singular or plural or part of a word) but no match on tracts is specified, just block groups are returned. Non-matching elements are ignored (e.g., sumlevel=c('bg', 'tracs', 'block') will return block groups but neither tracts (because of the typo) nor blocks (not available in ACS), with no warning – No warning is given if sumlevel is set to a list of elements where some are not recognized as matches to bg or tract, as long as one or more match bg, tracts, or both (or variants as already noted).

`vars` Optional, default is 'all' (in which case all variables from each table will be returned unless otherwise specified – see below). This parameter specifies whether to pause and ask the user about which variables are needed in an interactive session in R. This gives the user a chance to prepare the file "variables needed.csv" (or just ensure it is ready), or to edit and save "variables needed.csv" within a window in the default editor used by R (the user is asked which of these is preferred). If vars is 'ask', the function just looks in `data.folder` for a file called "variables needed.csv" that, if used, must specify which variables to keep from each table. The format of that file should be the same as is found in the file "variables needed template.csv" created by this function – keeping the letter "Y" in the column named "keep" indicates the given variable is needed. A blank or "N" or other non-Y specifies the variable should be dropped and not returned by get.acs(). If the "variables needed.csv" file is not found, however, this function looks for and uses the file called "variables needed template.csv" which is written by this function and specifies all of the variables from each table, so all variables will be retained and returned by get.acs().

`varsfile` See help for `set.needed` for details. Optional name of file that can be used to specify which variables are needed from specified tables. If varsfile is specified, parameter vars is ignored, and the function just looks in folder for file called filename, e.g., "variables needed.csv" that should specify which variables to keep from each table.

`new.geo` Default is TRUE. If FALSE, just uses existing downloaded geo file if possible. If TRUE, forced to download geo file even if already done previously.

write.files	Default is FALSE, but if TRUE then data-related csv files are saved locally – Saves longnames, full fieldnames as csv file, in working directory.
save.files	Default is FALSE, but if TRUE then various intermediate image files are saved as .RData files locally in working directory.
write.acspkg	Default is FALSE. If TRUE, saves csv file of tracts and file of block groups, for each of the tables, in a format that the Census Bureau American Fact Finder provides as downloadable tables. That format can be easily read in by the very useful **acs** package.
testing	Default is FALSE, but if TRUE more information is shown on progress, using cat() and while downloading, and more files (csv) are saved in working directory. But see silent parameter.
noEditOnMac	FALSE by default. If TRUE, do not pause to allow edit() to define which variables needed from each table, when on Mac OSX, even if vars=TRUE. Allows you to avoid problem in RStudio if X11 not installed.
silent	Optional logical, default is FALSE. Should progress updates be shown (sent to standard out, like the screen).
save.log	Optional logical, default is TRUE. Should log file be saved in output.path folder
filename.log	Optional name (without extension) for a log file, which gets date and time and .txt extension appended to it. Default is "log"

Details

The United States Census Bureau provides detailed demographic data by US Census tract and block group in the American Community Survey (ACS) 5-year summary file via their FTP site. For those interested in block group or tract data, Census Bureau tools tend to focus on obtaining data one state at a time rather than for the entire US at once. This function lets a user specify (tables and) variables needed. This will look up what sequence files contain those tables. Using a table of variables for those selected tables, a user can specify variables or tables to drop by putting x or anything other than "Y" in the column specifying needed variables.

For ACS documentation, see for example:

http://www.census.gov/programs-surveys/acs.html

Value

By default, returns a list of ACS data tables and information about them, with these elements in the list:
bg, tracts, headers, and info. The headers and info elements are data.frames providing metadata such as short and long field names. The same column names are found in x$info and x$headers, but headers has more rows. The info table just provides information about each data variable in the estimates tables. The headers table provides similar information but made to match the bg or tract format, so the headers table has as many rows as bg or tracts has columns – enough for the estimates and MOE fields, and the basic fields such as FIPS. The info data.frame can look like this, for example:

```
  'data.frame': xxxx obs. of  9 variables:
 $ table.ID       : chr  "B01001" "B01001" "B01001" "B01001" ...
 $ line           : num  1 2 3 4 5 6 7 8 9 10 ...
```

```
 $ shortname      : chr  "B01001.001" "B01001.002" "B01001.003" "B01001.004" ...
 $ longname       : chr  "Total:" "Male:" "Under 5 years" "5 to 9 years" ...
 $ table.title    : chr  "SEX BY AGE" "SEX BY AGE" "SEX BY AGE" "SEX BY AGE" ...
 $ universe     : chr "Universe: Total population" "Universe: Total population" "Universe: Total p
 $ subject        : chr  "Age-Sex" "Age-Sex" "Age-Sex" "Age-Sex" ...
 $ longname2      : chr  "Total" "Male" "Under5years" "5to9years" ...
 $ longname.unique: chr "Total:|SEX BY AGE" "Male:|SEX BY AGE" "Under 5 years|SEX BY AGE" "5 to 9 yea
```

Note

#' ################################### ADDITIONAL NOTES ###################################
SEE CENSUS ACS DOCUMENTATION ON NON-NUMERIC FIELDS IN ESTIMATE AND
MOE FILES.

Note relevant sequence numbers change over time.
#############

##
OLDER NOTES on where to obtain ACS data - sources for downloads of summary file data
##

FOR ACS SUMMARY FILE DOCUMENTATION, SEE
https://www.census.gov/acs/www/data_documentation/summary_file/

LARGER THAN NECESSARY:

*** all states and all tables in one huge tar.gz file (plus a zip of all geography codes)
ftp://ftp.census.gov/acs2011_5yr/summaryfile//2007-2011_ACSSF_All_In_2_Giant_Files(Experienced-U
ftp://ftp.census.gov/acs2011_5yr/summaryfile//2007-2011_ACSSF_All_In_2_Giant_Files(Experienced-U
/2011_ACS_Geography_Files.zip
2011_ACS_Geography_Files.zip #
Tracts_Block_Groups_Only.tar.gz # (THIS HAS ALL THE SEQUENCE FILES FOR EACH STATE
IN ONE HUGE ZIP FOLDER)

or

MORE FOCUSED DOWNLOADS:

*** one zip file per sequence file PER STATE: (plus csv of geography codes)

http://www2.census.gov/acs2011_5yr/summaryfile/2007-2011_ACSSF_By_State_By_Sequence_
Table_Subset/DistrictOfColumbia/Tracts_Block_Groups_Only/

20115dc0002000.zip (Which has the e file and m file for this sequence file in this state)
g20115dc.csv (which lets you link FIPS to data)
so this would mean 50+ * about 7 sequence files? = about 350 zip files?? each with 2 text files. Plus 50+ geo csv files. so downloading about 400 files and expanding to about 750 files, and joining into one big file.

OR

PREJOINED TO TIGER BLOCK GROUP BOUNDARIES SHAPEFILES/ GEODATABASES - ONE PER STATE HAS SEVERAL TABLES
http://www.census.gov/geo/maps-data/data/tiger-data.html
BUT NOT one file per sequence file FOR ALL STATES AT ONCE.
Estimates & margin of error (MOE), (ONCE UNZIPPED), and GEOgraphies (not zipped) are in 3 separate files.

also, data for entire US for one seq file at a time, but not tract/bg – just county and larger? – is here, e.g.: ftp://ftp.census.gov/acs2011_1yr/summaryfile//2011_ACSSF_By_State_By_Sequence_Table_Subset/UnitedStates/20111us0001000.zip
GEO files:
Note the US file is not bg/tract level: geo for whole US at once doesn't have tracts and BGs ftp://ftp.census.gov/acs2011_1yr/summaryfile//2011_ACSSF_By_State_By_Sequence_Table_Subset/UnitedStates/g20111us.csv

OTHER SOURCES include

- **tidycensus** package for R - uses API, requires a key, very useful for modest numbers of Census units rather than every block group in US
- **acs** package for R - uses API, requires a key, very useful for modest numbers of Census units rather than every block group in US
- http://www.NHGIS.org - (and see nhgis) very useful for block group (or tract/county/state/US) datasets
- DataFerrett (http://dataferrett.census.gov/AboutDatasets/ACS.html) – not all tracts in US at once
- American Fact Finder (http://www.census.gov/acs/www/data/data-tables-and-tools/american-factfinder/) (not block groups for ACS SF, and the tracts are not for the whole US at once)
- ESRI - commercial
- Geolytics - commercial
- etc.

See Also

acs package, which allows you to download and work with ACS data (using the API and your own key). To get the tables and variables used in EJSCREEN, see ejscreen.download. Also see nhgis which parses any files manually downloaded from NHGIS.org

Examples

```
##### Basic info on ACS tables:
cbind(table(lookup.acs2020$Subject.Area))
```

```
                                                                                table.title table.ID
                                                                                  SEX BY AGE   B01001
                                                                  HISPANIC OR LATINO ORIGIN BY RACE   B03002
                                       SEX BY EDUCATIONAL ATTAINMENT FOR THE POPULATION 25 YEARS AND OVER  B15002

AGE BY LANGUAGE SPOKEN AT HOME BY ABILITY TO SPEAK ENGLISH FOR THE POPULATION 5 YEARS AND OVER  B16004
                                         HOUSEHOLD LANGUAGE BY HOUSEHOLD LIMITED ENGLISH SPEAKING STATUS  C16002

                                                RATIO OF INCOME TO POVERTY LEVEL IN THE PAST 12 MONTHS  C17002
                                               EMPLOYMENT STATUS FOR THE POPULATION 16 YEARS AND OVER  B23025
                                                                                    YEAR STRUCTURE BUILT  B25034   #
```

```
 # from the census website, see USA ACS tables:
mytables <- c("B01001", "B03002", "B15002", 'B23025', "B25034", "C16002", "C17002") # EJScreen-related
yr <- 2020 # ACS 5YR Summary File 2016-2020
geos <- '0100000US_0400000US72' # US (just 50 States and DC) and PR:
https://data.census.gov/cedsci/table?q=acs%20B01001%20B03002%20B15002%20B23025%20B25034%20C16002%20C17002
myurl <- paste0('https://data.census.gov/cedsci/table?q=acs%20', paste(mytables, collapse= '%20'), '&g=', ge
# browseURL(myurl)
```

```
1    B03002    1 B03002.001                                                    Total:
2    B03002    2 B03002.002     Not Hispanic or Latino:
  3    B03002    3 B03002.003                                                White alone
  4    B03002    4 B03002.004                            Black or African American alone
  5    B03002    5 B03002.005                  American Indian and Alaska Native alone
  6    B03002    6 B03002.006                                                Asian alone
  7    B03002    7 B03002.007           Native Hawaiian and Other Pacific Islander alone
  8    B03002    8 B03002.008                                        Some other race alone
  9    B03002    9 B03002.009                                            Two or more races
  12   B03002   12 B03002.012   Hispanic or Latino
```

```
## Not run:
 t( get.table.info('B01001', end.year = acsdefaultendyearhere_func()) )
 t( get.table.info(c('B17001', 'C17002') ) )
 get.field.info('C17002')
 ##### Data for just DC & DE, just two tables:
 outsmall <- get.acs(tables = c('B01001', 'C17002'), mystates=c('dc','de'),
 end.year = acsdefaultendyearhere_func(), base.path = '~/Downloads', write.files = T, new.geo = FALSE)
 summary(outsmall)
 t(outsmall$info[1, ])
 t(outsmall$bg[1, ])

  ## ENTIRE USA -- DOWNLOAD AND PARSE -- TAKES A COUPLE OF MINUTES for one table:
  acs_2014_2018_B01001_vars_bg_and_tract <- get.acs(
    base.path='~/Downloads', end.year='2018', write.files = TRUE, new.geo = FALSE)

 ##########################################################################
 ##### Data for just DC & DE, just the default pop count table:
out <- get.acs(mystates=c('dc','de'), end.year = acsdefaultendyearhere_func(), new.geo = FALSE)
 names(out$bg); cat('\n\n'); head(out$info)
head(t(rbind(id=out$headers$table.ID, long=out$headers$longname, univ=out$headers$universe,
    subj=out$headers$subject,  out$bg[1:2,]) ), 15)
 cbind(longname=out$info$longname,
```

```
      total=colSums(out$bg[ , names(out$bg) %in% out$info$shortname ]))
  ### to see data on 2 places, 1 per column, with short and long field names
  cbind( out$headers$longname, t(out$bg[1:2, ]) )
  ### to see 7 places, 1 per row, with short and long field name as header
  head( rbind(out$headers$longname, out$bg) )[,1:7]
  ##### just 2 tables for just Delaware
  out <- get.acs(mystates='de', tables=c('B01001', 'C17002'))
  summary(out); head(out$info); head(out$bg)
  ##### uses all EJSCREEN defaults and the specified folders:
  out <- get.acs(base.path='~', data.path='~/ACStemp', output.path='~/ACSresults')
  summary(out); head(out$info); head(out$bg)
 ##### all tables needed for EJSCREEN, plus 'B16001', # b16001 has more details on specific languages spoken
    with variables specified in 'variables needed.csv', all states and DC and PR:
  out <- get.acs(tables=c('ejscreen', 'B16001'))
  summary(out); head(out$info); head(out$bg)

## End(Not run)
```

`get.acs.all` *get ACS summaryfile specified tables for entire USA, all blockgroups or all tracts*

Description

Uses Census Bureau new format files that offer the full USA in 1 file per table, via FTP or https. Does not get Puerto Rico (PR) as coded. Note API provides better list of variable names, in order as found on summary files, than documentation does.

Usage

```
get.acs.all(
  tables = "B01001",
  end.year = 2020,
  dataset = "5",
  sumlevel = 150,
  output.path = file.path("~", "acsoutput")
)
```

Arguments

`tables`	'ejscreen' or else one or more table codes like 'b01001' or c("B01001", "B03002")
`end.year`	2020 or another available year
`dataset`	default is 5, for 5-year summary file data like 2016-2020 ACS
`sumlevel`	default is 150 for blockgroups. 140 is tracts.
`output.path`	folder / path where to save the downloaded files

Value

list of data.frames, one per table requested

get.bg — *Get just the Census block group part of existing ACS data*

Description

Helper function to return just the block group resolution part of a dataset in get.acs

Usage

```
get.bg(merged.tables.mine)
```

Arguments

merged.tables.mine	Required set of tables in format used by get.acs

Value

subset of the inputs, same format

See Also

get.acs, get.tracts

get.datafile.prefix — *Get first part of ACS datafile name.*

Description

Get the first part of the datafile name for the ACS 5-year summary file datafiles on the US Census Bureau FTP site.

Usage

```
get.datafile.prefix(end.year = acsdefaultendyearhere_func())
```

Arguments

end.year	Optional character, such as "2012", specifying last year of 5-year summary file data.

See Also

get.acs, datafile, geofile, get.zipfile.prefix

`get.field.info` *Get short and long field names etc for ACS tables*

Description

Get info on tables from US Census Bureau for American Community Survey 5-year summary file.

Usage

```
get.field.info(
  tables,
  end.year = acsdefaultendyearhere_func(),
  table.info.only = FALSE,
  moe = FALSE,
  basic = FALSE,
  silent = FALSE
)
```

Arguments

`tables`	Required vector of tables such as "B01001"
`end.year`	Last year of 5-year summary file such as '2018'
`table.info.only`	FALSE by default. If TRUE, only return info about the table(s), not variables in table(s).
`moe`	FALSE by default. Margin of error variables also included if TRUE, but their names are identical to those of estimates fields other than being MOE instead of estimate.
`basic`	FALSE by default. If TRUE, a very limited subset of the info is returned (just variable number and name). This parameter is ignored if table.info.only=TRUE
`silent`	Optional logical, FALSE by default. Whether to send progress info to standard output (like the screen)

Details

Uses `get.lookup.acs` but for latest version could just use `data(lookup.acs)`

Value

data.frame of information about each table and each variable in table:
Value returned is data.frame of info about each table and also each variable in the table
e.g., longname2 which is version where no spaces or colons or escaped quotation marks etc.

See Also

`get.acs`, `get.table.info`, and `get.field.info`

Examples

```
## Not run:
  finfo <- get.field.info(c('B17020A', 'B17020H'))
  cbind(
   names(tracts), substr(finfo$longname.unique[match(names(tracts), finfo$shortname)], 1, 100)
  )
  foundnames <- names(tracts)[names(tracts) %in% finfo$shortname]
  cbind(
    foundnames, substr(finfo$longname.unique[match(foundnames, finfo$shortname)], 1, 100)
  )

## End(Not run)
```

`get.lookup.acs` *Get Information about ACS 5-Year Summary File Tables*

Description

Get lookup table of information on American Community Survey (ACS) tables, from the Census Bureau, namely which sequence files on the FTP site contain which tables and which variables. NOTE: This uses lazy loading from datasets in the package.

Usage

```
get.lookup.acs(end.year = acsdefaultendyearhere_func())
```

Arguments

`end.year`	Character, optional, like '2020', which specifies the 2016-2020 dataset. Defines which 5-year summary file to use, based on end-year. Note: Function stops with error if given end.year is not yet added to this package or is too old.

Details

Now folders are or were like: https://www2.census.gov/programs-surveys/acs/summary_file/2017/data/5_year_seq_by_s
https://www2.census.gov/programs-surveys/acs/summary_file/2017/documentation/user_tools/

Value

returns a data.frame `dim(lookup.acs2020)` `[1]` `30326` `9` `names(lookup.acs2020)` `[1]` `"File.ID"` `"Table.ID"` `"Sequence.Number"` `[4]` `"Line.Number"` `"Start.Position"` `"Total.Cells.in.Table"` `[7]` `"Total.Cells.in.Sequence"` `"Table.Title"` `"Subject.Area"`

See Also

`get.table.info` and `get.field.info`.
Also see `acs.lookup` which does something similar but is more flexible & robust. Also see `download.lookup.acs` to download the file from the Census FTP site. Also see `data(lookup.acs2020)` and similar data for other years. Also see `get.acs`, `get.lookup.file.name`, `get.url.prefix.lookup.table`

Examples

```
## Not run:
lookup.acs <- get.lookup.acs()

## End(Not run)
```

`get.lookup.file.name` *Get name of Census file with Sequence and Table Numbers for ACS 5-year summary file data*

Description

Returns name of text file provided by US Census Bureau, such as Sequence_Number_and_Table_Number_Lookup.txt, which provides the sequence numbers (file numbers) and table numbers for data in the American Community Survey (ACS) 5-year summary file.

Usage

```
get.lookup.file.name(end.year = acsdefaultendyearhere_func())
```

Arguments

`end.year`	Not yet implemented, but will be optional end year for 5-year summary file, as character

Value

Returns character element that is name of file such as "Sequence_Number_and_Table_Number_Lookup.txt"

See Also

`get.acs`, `get.lookup.acs`, `get.url.prefix.lookup.table`. Also see `data(lookup.acs)`.

`get.read.geo` *Download (if necessary) and merge GEO files for ACS*

Description

Returns a data.frame of all states merged geo info and saves geo.RData to working directory.

Usage

```
get.read.geo(
  mystates,
  new.geo = FALSE,
  folder = getwd(),
  end.year = acsdefaultendyearhere_func(),
  testing = FALSE,
  silent = FALSE
)
```

Arguments

mystates	Character vector of 2-character state abbreviations, required.
new.geo	Logical value, optional, FALSE by default. If FALSE, if geo exists in memory don't download and parse again.
folder	Defaults to current working directory.
end.year	optional character year to specify last year of 5-year summary file.
testing	Default to FALSE. If TRUE, provides info on progress of download.
silent	Default is FALSE.

Details

Uses download.geo() then read.geo(), and then does some cleanup.

Note that if this finds the geographic file in folder already, it will not download it again even if that file was corrupt. Read and compile geo data for entire USA with PR DC,
This takes some time for the entire USA:
2 minutes 48 seconds on
MacBook Pro 16 GB RAM, SSD, 2.7 GHz Intel Core i7
R version: R 3.0.2 GUI 1.62 Snow Leopard build (6558)
8 minutes 20 seconds on
R version 3.0.1
Dell Latitude E6400, x86 Family 6 Model 23 Stepping 6 GenuineIntel ~2260 Mhz,
Total Physical Memory 4,096.00 MB, Available Physical Memory 1.97 GB, Total Virtual Memory 2.00 GB, Available Virtual Memory 1.94 GB,
Microsoft Windows XP Professional
###############

Value

Returns a data.frame of all states geo info.
# FOR ACS 2008-2012, tract and block group counts:
table(geo$SUMLEVEL)
140 150
74001 220333

Remaining fields in geo:
"STUSAB" "SUMLEVEL" "GEOID" "FIPS" "KEY"
NOTE: do not really need GEOID or KEY.
GEOID is redundant, but might be useful for joining to shapefiles/ boundaries
Also, could specify here if "NAME" field from geo files should be dropped - it might be useful but takes lots of RAM and encoding of Spanish characters in Puerto Rico caused a problem in Mac OSX.
NOTE FROM CENSUS:
The ACS Summary File GEOID contains the necessary information to connect to the TIGER/Line Shapefiles, but it needs to be modified in order to exactly match up. Notice that the ACS GEOID, 05000US10001, contains the TIGER/Line GEOID string, 10001. In order to create an exact match of both GEOIDs, it is necessary to remove all of the characters before and including the letter S in the ACS Summary File. By removing these characters, the new GEOID in the ACS Summary File exactly matches the field GEOID in the TIGER/Line Shapefiles.

See Also

get.acs which uses this, and download.geo

---

get.table.info *Get field names etc for ACS tables*

Description

Get info on tables from US Census Bureau for American Community Survey 5-year summary file.

Usage

```
get.table.info(
  tables,
  end.year = acsdefaultendyearhere_func(),
  table.info.only = TRUE,
  moe = FALSE
)
```

Arguments

tables	Required vector of tables such as "B01001"
end.year	Last year of 5-year summary file such as '2012'
table.info.only	TRUE by default. See get.field.info
moe	FALSE by default. If TRUE, returns MOE versions of field names and descriptions.

Details

Wrapper for get.table.info2 which is a wrapper for get.field.info

Value

data.frame of information about each table and each variable in table:
Table.ID, Line.Number, Table.Title, table.var, varname2

Value returned is data.frame of info about each table and also each variable in the table:

Table.ID Line.Number Table.Title table.var varname2
7 B01001 NA SEX BY AGE <NA> SEXBYAGE
8 B01001 NA Universe: Total population <NA> UniverseTotalpopulation
9 B01001 1 Total: B01001.001 Total
10 B01001 2 Male: B01001.002 Male

See Also

get.acs, get.table.info, and get.field.info

`get.table.info2` *Get field names etc for ACS tables*

Description

Get info on tables from US Census Bureau for American Community Survey 5-year summary file.

Usage

```
get.table.info2(
  tables,
  end.year = acsdefaultendyearhere_func(),
  table.info.only = TRUE,
  moe = FALSE
)
```

Arguments

`tables`	Required vector of tables such as "B01001"
`end.year`	Last year of 5-year summary file such as '2018'
`table.info.only`	TRUE by default. See `get.field.info`
`moe`	FALSE by default. If TRUE, returns MOE versions of field names and descriptions.

Details

Wrapper for `get.field.info`

Value

data.frame of information about each table and each variable in table:
Table.ID, Line.Number, Table.Title, table.var, varname2

Value returned is data.frame of info about each table and also each variable in the table:

Table.ID Line.Number Table.Title table.var varname2
7 B01001 NA SEX BY AGE <NA> SEXBYAGE
8 B01001 NA Universe: Total population <NA> UniverseTotalpopulation
9 B01001 1 Total: B01001.001 Total
10 B01001 2 Male: B01001.002 Male

See Also

`get.acs`, `get.table.info`, and `get.field.info`

get.tracts — *Get just the Census tracts part of existing ACS data*

Description

Helper function to return just the tract resolution part of a dataset in get.acs

Usage

```
get.tracts(merged.tables.mine)
```

Arguments

merged.tables.mine	Required set of tables in format used by get.acs

Value

subset of the inputs, same format

See Also

get.acs, get.bg

get.url.prefix — *Get URL prefix for FTP site folder(s) with ACS 5-year summary file data*

Description

Returns part of URL of folders (on Census Bureau FTP site) with zip file(s) based on end year.

Usage

```
get.url.prefix(end.year = acsdefaultendyearhere_func())
```

Arguments

end.year	Optional end year for 5-year summary file, as character, but ignored if url.prefix is specified

Value

Returns character vector that is first part of URL such as "ftp://ftp.census.gov/acs2012_5yr/summaryfile/2008-2012_ACSSF_By_State_By_Sequence_Table_Subset"

See Also

get.acs, url.to.find.zipfile, download.geo

get.url.prefix.lookup.table *Get first part of name(s) of URL(s) for ACS 5-year summary file data*

Description

Returns first part of URL(s) of folders (on Census Bureau FTP site) with zip file(s) based on end year.

Usage

```
get.url.prefix.lookup.table(end.year = acsdefaultendyearhere_func())
```

Arguments

end.year	Optional end year for 5-year summary file, as character, like '2018'

Value

Returns character element that is first part of URL such as "ftp://ftp.census.gov/acs2012_5yr/summaryfile/"

See Also

get.acs, get.lookup.acs, get.lookup.file.name

get.zipfile.prefix *Get first part of ACS zip file name.*

Description

Get the first part of the zipfile name for the ACS 5-year summary file datafiles on the US Census Bureau FTP site.

Usage

```
get.zipfile.prefix(end.year = acsdefaultendyearhere_func())
```

Arguments

end.year	Optional character, specifying last year of 5-year summary file data.

See Also

get.acs, get.datafile.prefix, datafile, geofile, get.zipfile.prefix

getseqnumsviafilenames *Infer Sequence file numbers based on ACS 5-yr filenames in folder*

Description

Helper function to look for unzipped csv files of estimates for American Community Survey (ACS) 5-year summary file data obtained from US Census FTP site, based on pattern matching, and infer seqfile numbers based on those filenames.

Usage

```
getseqnumsviafilenames(folder = getwd())
```

Arguments

folder	Default is current working directory.

Value

Returns a character vector of unique sequence file numbers

See Also

read.concat.states which uses this

getstatesviafilenames *Infer US States based on ACS 5-yr filenames in folder*

Description

Helper function to look for unzipped csv files of estimates for American Community Survey (ACS) 5-year summary file data obtained from US Census FTP site, based on pattern matching, and infer State abbreviations based on those filenames.

Usage

```
getstatesviafilenames(folder = getwd())
```

Arguments

folder	Default is current working directory.

Value

Returns a vector of unique upper case US State abbreviations

See Also

read.concat.states which uses this

gettablesviaseqnums *Look up ACS table IDs of tables in given sequence number files*

Description

Helper function to look for which American Community Survey (ACS) tables are in given sequence files as obtained from the US Census FTP site.

Usage

```
gettablesviaseqnums(x, end.year = acsdefaultendyearhere_func())
```

Arguments

x	Required character vector of table IDs such as "B01001"
end.year	Optional character variable providing the end year of the 5-year ACS survey, to ensure the proper sequence file to table ID matching is used.

Value

Returns a character vector of table IDs such as "B01001"

See Also

read.concat.states which uses this, and get.lookup.acs which is used by this

join.geo.to.tablist *Join US ACS data and US geo files on FIPS*

Description

Read the processed csv files of estimates and MOE (margin of error) for American Community Survey (ACS) 5-year summary file data obtained from US Census FTP site, and join with geographic information from geo file.

Usage

```
join.geo.to.tablist(
  mygeo,
  my.list.of.tables,
  save.csv = FALSE,
  sumlevel = "both",
  folder = getwd(),
  testing = FALSE,
  end.year = acsdefaultendyearhere_func()
)
```

Arguments

mygeo	Required geo file. See get.acs and get.read.geo
my.list.of.tables	List of data tables resulting from prior steps in get.acs
save.csv	FALSE by default. Specifies whether to save each data table as csv format file.
sumlevel	Default is "both", specifies if "tracts" or "blockgroups" or "both" should be used.
folder	Default is current working directory.
testing	Default is FALSE. If TRUE, prints more information.
end.year	Default is "" – used in naming file if save.csv=TRUE

Value

Returns a list of data.frames, where each element of the list is one ACS table, such as table B01001.

See Also

get.acs and get.read.geo

---

list_files_ftp *list_files_ftp Attempt to get list of filenames on an FTP site*

Description

list_files_ftp Attempt to get list of filenames on an FTP site

Usage

```
list_files_ftp(
  url,
  credentials = "",
  sleep = NA,
  sort = FALSE,
  verbose = FALSE
)
```

Arguments

url	URL
credentials	empty
sleep	NA
sort	FALSE
verbose	FALSE

Value

file names

list_files_ftp_worker *list_files_ftp_worker Attempt to get list of filenames on an FTP site. See list_files_ftp() Uses RCurl::getURL(url, dirlistonly = TRUE)*

Description

list_files_ftp_worker Attempt to get list of filenames on an FTP site. See list_files_ftp() Uses RCurl::getURL(url, dirlistonly = TRUE)

Usage

```
list_files_ftp_worker(url, credentials, sleep, verbose)
```

Arguments

url	URL
credentials	?
sleep	?
verbose	?

Value

list of file names

lookup.acs *LATEST ACS_5yr_Seq_Table_Number_Lookup.txt for ACS dataset*

Description

This data set provides information about variables in tables forming the American Community Survey (ACS) 5-year summary file.

This data set provides information about variables in tables forming the American Community Survey (ACS) 5-year summary file.

Format

A data.frame with these fields:
'data.frame': 29501 obs. of 9 variables:

- $ File.ID : chr "ACSSF" ...
- $ File.ID : chr "ACSSF" "ACSSF" "ACSSF" "ACSSF" ...
- $ Table.ID : chr "B01001" "B01001" "B01001" "B01001" ...
- $ Sequence.Number : chr "0001" "0001" "0001" "0001" ...
- $ Line.Number : num NA NA 1 2 3 4 5 6 7 8 ...
- $ Start.Position : num 7 NA NA NA NA NA NA NA NA NA ...
- $ Total.Cells.in.Table : chr "49 CELLS" NA NA NA ...
- $ Total.Cells.in.Sequence: num NA NA NA NA NA NA NA NA NA NA ...

- $ Table.Title : chr "SEX BY AGE" "Universe: Total population" "Total:" "Male:" ...
- $ Subject.Area : chr "Age-Sex" NA NA NA ...

A data.frame

Details

See attributes(lookup.acs) to check vintage of dataset. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1. `dim(lookup.acs2020)` [1] 30326 9 `names(lookup.acs2020)` [1] "File.ID" "Table.ID" "Sequence.Number" [4] "Line.Number" "Start.Position" "Total.Cells.in.Table" [7] "Total.Cells.in.Sequence" "Table.Title" "Subject.Area"

```
 # example of how data was imported after download


setwd('./inst/')
download.file('https://www2.census.gov/programs-surveys/acs/summary_file/2020/documentation/user
              destfile = 'ACS_5yr_Seq_Table_Number_Lookup.txt')
lookup.acs <- readr::read_csv('ACS_5yr_Seq_Table_Number_Lookup.txt')
lookup.acs <- data.frame(lookup.acs, stringsAsFactors = FALSE) # so it is not a tibble

metadata <- list(ejscreen_releasedate = 'late 2022', ejscreen_version = '2.1', ACS_version = '2016-2
attributes(lookup.acs) <- c(attributes(lookup.acs), metadata)

# save(lookup.acs, file = './data/lookup.acs.rdata') # or
usethis::use_data(lookup.acs)
```

See attributes(lookup.acs) to check vintage of dataset. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1. `dim(lookup.acs2020)` [1] 30326 9 `names(lookup.acs2020)` [1] "File.ID" "Table.ID" "Sequence.Number" [4] "Line.Number" "Start.Position" "Total.Cells.in.Table" [7] "Total.Cells.in.Sequence" "Table.Title" "Subject.Area"

```
 # example of how data was imported after download


setwd('./inst/')
download.file('https://www2.census.gov/programs-surveys/acs/summary_file/2020/documentation/user
              destfile = 'ACS_5yr_Seq_Table_Number_Lookup.txt')
lookup.acs <- readr::read_csv('ACS_5yr_Seq_Table_Number_Lookup.txt')
lookup.acs <- data.frame(lookup.acs, stringsAsFactors = FALSE) # so it is not a tibble

metadata <- list(ejscreen_releasedate = 'late 2022', ejscreen_version = '2.1', ACS_version = '2016-2
attributes(lookup.acs) <- c(attributes(lookup.acs), metadata)

# save(lookup.acs, file = './data/lookup.acs.rdata') # or
usethis::use_data(lookup.acs)
```

See Also

acs.lookup which does something similar but is more flexible & robust. Also see get.lookup.acs which downloads these files. Also see get.acs.

acs.lookup which does something similar but is more flexible & robust. Also see get.lookup.acs which downloads these files. Also see get.acs.

Examples

```
## Not run:
data(lookup.acs2019, package='ACSdownload')
# or
lookup.acs <- ACSdownload::get.lookup.acs(2019)
# or related info from
acs::acs.lookup()

## End(Not run)
```

---

lookup.acs2018 *ACS_5yr_Seq_Table_Number_Lookup.txt for ACS dataset*

Description

This data set provides information about variables in tables forming the American Community Survey (ACS) 5-year summary file. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1, so 2014-2018 would be available by Dec 2019. @details # example of how data was imported after download # # setwd('~/R/ACSdownload/inst/') # download.file('https://www2.census.gov/programs-surveys/acs/summary_file/2018/documentation/user_tools/ACS_5yr
destfile = 'ACS_5yr_Seq_Table_Number_Lookup-2018.txt') # lookup.acs2018 <- readr::read_csv('ACS_5yr_Seq_Table
2018.txt') # lookup.acs2018 <- data.frame(lookup.acs2018, stringsAsFactors = FALSE) # setwd('~/R/ACSdownload/data
save(lookup.acs2018, file = 'lookup.acs2018.rdata')

Format

A data.frame with these fields:
'data.frame': approx 25074 obs. of approx 8 variables:

- $ Table.ID : chr "B00001" "B00001" "B00001" "B00002" ...
- $ Sequence.Number : chr "0001" "0001" "0001" "0001" ...
- $ Line.Number : num NA NA 1 NA NA 1 NA NA 1 2 ...
- $ Start.Position : num 7 NA NA 8 NA NA 7 NA NA NA ...
- $ Total.Cells.in.Table : chr "1 CELL" "" "" "1 CELL" ...
- $ Total.Cells.in.Sequence: num NA NA NA 2 NA NA NA NA NA NA ...
- $ Table.Title : chr "UNWEIGHTED SAMPLE COUNT OF THE POPULATION" "Universe: Total population" "Total" "UNWEIGHTED SAMPLE HOUSING UNITS" ...
- $ Subject.Area : chr "Unweighted Count" "" "" "Unweighted Count" ...

See Also

acs.lookup which does something similar but is more flexible & robust. Also see get.lookup.acs which downloads these files. Also see get.acs.

Examples

```
## Not run:
data(lookup.acs2016, package='ACSdownload')
# or
lookup.acs <- ACSdownload::get.lookup.acs(2018)
# or related info from
acs::acs.lookup()

## End(Not run)
```

`lookup.acs2020` *ACS_5yr_Seq_Table_Number_Lookup.txt for ACS dataset*

Description

This data set provides information about variables in tables forming the American Community Survey (ACS) 5-year summary file. The data and documentation for the 5 years ending in year X is typically available by December of the year X+1, so 2016-2020 was planned for Dec 2021 but actually out March 2022.

```
# example of how data was imported after download
#
# setwd('./inst/')
# download.file('https://www2.census.gov/programs-surveys/acs/summary_file/2019/documentation/us
# lookup.acs2019 <- readr::read_csv('ACS_5yr_Seq_Table_Number_Lookup.txt')
# lookup.acs2019 <- data.frame(lookup.acs2019, stringsAsFactors = FALSE)
# save(lookup.acs2019, file = './data/lookup.acs2019.rdata') # or usethis::use_data(lookup.acs201
```

Format

A data.frame
`dim(lookup.acs2020)` `[1] 30326 9` `names(lookup.acs2020)` `[1] "File.ID" "Table.ID" "Sequence.Number"`
`[4] "Line.Number" "Start.Position" "Total.Cells.in.Table" [7] "Total.Cells.in.Sequence"`
`"Table.Title" "Subject.Area"`

See Also

`acs.lookup` which does something similar but is more flexible & robust. Also see `get.lookup.acs` which downloads these files. Also see `get.acs`.

Examples

```
## Not run:
data(lookup.acs2019, package='ACSdownload')
# or
lookup.acs <- ACSdownload::get.lookup.acs(2019)
# or related info from
acs::acs.lookup()

## End(Not run)
```

`merge_tables` *Concatenate several ACS data tables into one big table*

Description

Concatenate several ACS data tables into one big table

Usage

```
merge_tables(my.list.of.tables)
```

Arguments

`my.list.of.tables`
: Required list of data tables from prior steps in `get.acs`

Value

Returns one big data.frame with all columns of all input tables

See Also

`get.acs`

`nhgis` *Read and Parse NHGIS.org ACS Data Files and Codebooks - not tested or updated*

Description

Read downloaded and unzipped csv and txt files obtained from NHGIS.org, with US Census Bureau data from the American Community Survey (ACS).

Needs to be updated for 2022 formats, etc.

Usage

```
nhgis(
  base.path = getwd(),
  code.dir = file.path(base.path, "nhgiscode"),
  data.dir = file.path(base.path, "nhgisdata"),
  silent = FALSE,
  savefiles = FALSE
)
```

Arguments

base.path	Optional base path, default is getwd()
code.dir	Optional path where extra code is. Not used.
data.dir	Optional path where data files are stored and output could be saved. Default is nhgiscode folder under base.path
silent	Optional, FALSE by default, whether to print info about progress, filenames found, etc.
savefiles	Optional, FALSE by default, whether to save .RData and maybe csv files of output returned.

Details

This was designed to read and parse csv and txt files obtained from NHGIS.org and already unzipped in a local folder. It only reads one set of files at a time, meaning the data and codebook files all have to be for the same set of ACS tables (a single NHGIS query) (but can be a separate data & codebook file pair for each spatial resolution like county, state, etc.) Obtaining NHGIS.org data requires an account at https://data2.nhgis.org/main, https://www.nhgis.org Data can be downloaded by selecting, for example,
tracts and block groups, all in US, acs2007-2011, and specifying the desired ACS Table(s).
Research using NHGIS data should cite it as:
Minnesota Population Center. National Historical Geographic Information System: Version 2.0. Minneapolis, MN: University of Minnesota 2011.

Documentation for NHGIS datasets is available here. Research using NHGIS data should cite it as: Steven Manson, Jonathan Schroeder, David Van Riper, Tracy Kugler, and Steven Ruggles. IPUMS National Historical Geographic Information System: Version 16.0 [dataset]. Minneapolis, MN: IPUMS. 2021. http://doi.org/10.18128/D050.V16.0 For policy briefs or articles in the popular press, we recommend that you cite the use of NHGIS data as follows: IPUMS NHGIS, University of Minnesota, www.nhgis.org

Value

Returns a named list, one element per summary level found (names are, e.g., 'us', 'states', etc.). Each summary level has a list of the following: data, contextfields, fields, tables, geolevel, years, dataset
For example:
summary(x[['us']]) Length Class Mode data 279 data.frame list contextfields 3 data.frame list fields 4 data.frame list tables 4 data.frame list geolevel 1 -none- character years 1 -none- character dataset 1 -none- character

See Also

nhgisread used by this function. Also, for other ways to obtain ACS data see get.acs

Examples

```
## Not run:
x <- nhgis(data.dir = '~/Desktop/nhgis0009_csv')
# save state data as csv
write.csv(x$states$data, file='statedata.csv', row.names = FALSE)
# Which geolevels were found (and what years)?
summary(x)
t(cbind(sapply(x, function(y) y[c('geolevel', 'years')])))
```

```
  summary(x[['counties']])
  # Which Census Bureau tables were found?
  x[['states']]$tables
  # See the data for one State
  t(x[['states']]$data[1, ])
  # How many counties are in each State?
  dat <- x[['counties']]$data
  cbind(sort(table(dat$STATE)))
  # How many counties have population > 1 million, for each State?
  cbind(sort(table(dat$STATE[dat$B01001.001 > 1E6])))

## End(Not run)
```

---

nhgisfind *Find NHGIS ACS Files on Disk*

Description

Look in specified path to find any downloaded and unzipped csv and txt files from NHGIS.ORG, with American Community Survey (ACS) data from the US Census Bureau.

Usage

```
nhgisfind(folder = getwd(), silent = FALSE)
```

Arguments

folder	Optional path to look in. Default is getwd().
silent	Optional, default is FALSE. Prints filenames if TRUE.

Details

This is designed to get a list of filenames that match the format of csv and txt files obtained from NHGIS.org and already unzipped in a local folder. Obtaining NHGIS.org data requires an account at https://data2.nhgis.org/main, https://www.nhgis.org Data can be downloaded by selecting, for example,
block groups, all in US, acs2007-2011, and specifying the desired ACS Table(s). Research using NHGIS data should cite it as:
Minnesota Population Center. National Historical Geographic Information System: Version 2.0. Minneapolis, MN: University of Minnesota 2011.

Value

A named list with datafiles= a vector of one or more filenames (estimates and also MOE files) and codebooks= a vector of one or more filenames. The function also prints the information unless silent=TRUE.

See Also

nhgis, nhgisread

nhgisfips *Assemble FIPS Code from State, County, Tract, Blockgroup Portions*

Description

Helper function for reading demographic data files downloaded from NHGIS.org

Usage

```
nhgisfips(
  x,
  validfields = c("STATEA", "COUNTYA", "TRACTA", "BLKGRPA"),
  fullname = c("FIPS.ST", "FIPS.COUNTY", "FIPS.TRACT", "FIPS.BG"),
  leadz = c(2, 3, 6, 1)
)
```

Arguments

x	Data.frame or matrix with appropriate colnames, containing portions of FIPS code in separate columns.
validfields	Optional, default is colnames used in datasets downloaded from NHGIS.org as of 8/2015 for ACS data. Defaults: c("STATEA", "COUNTYA", "TRACTA", "BLKGRPA")
fullname	Optional, default is based on default for validfields parameter: c("FIPS.ST", "FIPS.COUNTY", "FIPS.TRACT", "FIPS.BG"). Specifies colname for the output, which depends on how many cols of fips portions are in x.
leadz	Optional, default is based on default for validfields parameter: c(2, 3, 6, 1) Defines total number of characters in correctly formatted portions of FIPS, such as 2 for State FIPS (e.g., "01").

Details

This can also be used more generically in other contexts, by specifying appropriate parameters.

Value

A 1-column data.frame with same number of rows as x. Provides assembled FIPS for each row.

See Also

nhgis, nhgisread, clean.fips

Examples

```
## Not run:
x <- structure(list(STATEA = structure(c(2L, 3L, 1L),
.Label = c("10", "8", "9"), class = "factor"),
COUNTYA = structure(1:3, .Label = c("1", "10", "100"), class = "factor"),
TRACTA = structure(c(2L, 1L, 2L), .Label = c("000006", "123456"), class = "factor"),
BLKGRPA = c("1", "2", "3"), data = c(0, 0, 0)),
.Names = c("STATEA", "COUNTYA", "TRACTA", "BLKGRPA", "data"),
```

```
row.names = c(NA, -3L), class = "data.frame")
x
nhgisfips(x[ , 1, drop=FALSE])
nhgisfips(x[ , 1:2], fullname=c('stfips', 'countyfips', 'tractfips', 'bgfips'))
nhgisfips(x[ , 1:3])
nhgisfips(x[ , 1:4])
nhgisfips(x)

## End(Not run)
```

`nhgisread` *Read NHGIS.org ACS Data Files and Codebooks*

Description

Helper function used by `nhgis` to read downloaded and unzipped csv and txt files obtained from NHGIS.org, with US Census Bureau data from the American Community Survey (ACS).

Usage

```
nhgisread(
  datafile,
  codebookfile = gsub("\\.csv", "_codebook.txt", datafile),
  folder = getwd()
)
```

Arguments

`datafile`	Names of files
`codebookfile`	Optional name(s) of codebook files. Default is to infer from datafile
`folder`	Optional path where files are found. Default is getwd()

Value

Returns a named list: data, contextfields, fields, tables, geolevel, years, dataset

See Also

`nhgis` which uses this, `nhgisreadcodebook` for reading codebook files, `get.acs`, `get.datafile.prefix`, `datafile`, `geofile`, `get.zipfile.prefix`

nhgisreadcodebook *Read NHGIS.org ACS Codebook File*

Description

Helper function used by nhgis to read downloaded and unzipped codebook files obtained from NHGIS.org, for US Census Bureau data from the American Community Survey (ACS).

Usage

```
nhgisreadcodebook(codebookfile, folder = getwd())
```

Arguments

codebookfile	Name(s) of codebook file(s).
folder	Optional path where files are found. Default is getwd()

Value

Returns a named list: data, contextfields, fields, tables, geolevel, years, dataset

See Also

nhgis which uses this, nhgisread for reading datafiles, get.acs, get.datafile.prefix, datafile, geofile, get.zipfile.prefix

read.concat.states *Read and concatenate State files of ACS data*

Description

Read the unzipped csv files of estimates and MOE (margin of error) for American Community Survey (ACS) 5-year summary file data obtained from US Census FTP site. These State-specific csv files are combined into a single national result for each Census data table.

Usage

```
read.concat.states(
  tables,
  mystates,
  geo,
  needed,
  folder = getwd(),
  output.path,
  end.year = acsdefaultendyearhere_func(),
  save.files = TRUE,
  sumlevel = "both",
  testing = FALSE,
  dt = TRUE,
  silent = FALSE
)
```

Arguments

tables	Optional character vector of table numbers needed such as 'B01001', but default is all tables from each sequence file found.
mystates	Optional character vector of 2-character state abbreviations. Default is all states for which matching filenames are found in folder.
geo	Optional table of geographic identifiers that elsewhere would be merged with data here. If provided, it is used here to look up data file length, based on state abbrev's list. See get.read.geo If geo is not provided, the function still reads each file whatever its length.
needed	Optional data.frame specifying which variables to keep from each table. Default is to keep all. See set.needed and get.acs
folder	Default is current working directory. Specifies where the csv files are to be found.
output.path	Default is whatever the parameter folder is set to. Results as .RData files are saved here if save.files=TRUE.
end.year	Optional, specifies end year of 5-year summary file.
save.files	Default is TRUE, in which case it saves each resulting table/ data file on disk.
sumlevel	Default is "both". Specifies if "tracts" or "blockgroups" or "both" should be returned.
testing	Default is FALSE. If TRUE, prints filenames but does not unzip them, and prints more messages.
dt	Optional logical, TRUE by default, specifies whether data.table::fread should be used instead of read.csv
silent	Default is FALSE. Whether to send progress info to standard output.

Details

This can use read.csv (takes about 1 minute total for one table, all states, est and moe). It can use data.table::fread (default method), which is faster.

Value

Returns a list of data.frames, where each element of the list is one ACS table, such as table B01001.

See Also

get.acs

---

read.geo *Read and concatenate state geo files from Census ACS*

Description

Reads and merges geo files that have been obtained from the US Census Bureau FTP site for American Community Survey (ACS) data. It uses read.fwf() at least for older years. 2018 or 2019 might have csv not just txt format now?

Usage

```
read.geo(
  mystates,
  folder = getwd(),
  end.year = acsdefaultendyearhere_func(),
  silent = FALSE
)
```

Arguments

mystates	Character vector of one or more states/DC/PR, as 2-character state abbreviations. Default is all states/DC/PR.
folder	Optional path to where files are stored, defaults to getwd()
end.year	End year of 5-year data, like "2018"
silent	Default is FALSE. Whether to send progress info to standard output.

Details

Note that if this finds the geographic file in folder already, it will not download it again even if that file was corrupt. Extracts just block group (SUMLEVEL=150) and tract (SUMLEVEL=140) geo information (not county info., since data files used in this package lack county info.) The NAME field works on pc but mac can hit an error if trying to read the NAME field. Due to encoding? specifying encoding didn't help.
(name is very long and not essential)
Error in substring(x, first, last) :
invalid multibyte string at '<f1>onc<69>to Chapter; Navajo Nation Reservation and Off-Reservation Trust Land, AZ–NM–UT
Format of files is here: ftp://ftp.census.gov/acs2012_5yr/summaryfile/ACS_2008-2012_SF_Tech_Doc.pdf

Value

Returns a large data.frame of selected geographic information on all block groups and tracts in the specified states/DC/PR, with just these fields:
"STUSAB", "SUMLEVEL", "LOGRECNO", "STATE", "COUNTY", "TRACT", "BLKGRP", "GEOID"

See Also

get.acs, download.geo

Examples

```
 ## Not run:
  geo <- read.geo( c("dc", "de") )

## End(Not run)
```

`set.needed` *Specify which ACS Variables are Needed*

Description

Utility used by get.acs to help user specify which variables are needed. User can specify this in a file in the working directory, modifying "variables needed template.csv" that this function can create based on tables parameter, to create user-defined "variables needed.csv"

Usage

```
set.needed(
  tables,
  lookup.acs,
  vars = "all",
  varsfile,
  folder = getwd(),
  noEditOnMac = FALSE,
  end.year = acsdefaultendyearhere_func(),
  silent = TRUE,
  writefile = TRUE
)
```

Arguments

`tables`	Character vector, required. Specifies which ACS tables.
`lookup.acs`	Data.frame, optional. Defines which variables are in which tables. Output of `get.lookup.acs`
`vars`	Optional default is all, which means all fields from table unless varsfile specified. Specifies what variables to use from specified tables, and how to determine that. If varsfile is provided, vars is ignored (see parameter varsfile). If vars="ask", function will ask user about variables needed and allow specification in an interactive session.
`varsfile`	Optional name of file that can be used to specify which variables are needed from specified tables. If varsfile is specified, parameter vars is ignored, and the function just looks in folder for file called filename, e.g., "variables needed.csv" that should specify which variables to keep from each table. If not found in folder, then all variables from each table are used (same as if vars="all" and varsfile not specified). The format of that file should be the same as is found in the file "variables needed template.csv" created by this function. If the filename (e.g., "variables needed.csv" file) is not found, it looks for and uses the file called "variables needed template.csv" which is written by this function and specifies all of the variables from each table. The column called "keep" should have an upper or lowercase letter Y to indicate that row (variable) should be kept. Blanks or other values (even the word "yes") indicate the variable is not needed from that data table and it will be dropped.
`folder`	Optional path, default is getwd(), specifying where to save the csv files that define needed variables.

noEditOnMac	FALSE by default. If TRUE, do not pause to allow edit() when on Mac OSX, even if vars=TRUE. Allows you to avoid problem in RStudio if X11 not installed.
end.year	Optional, – specifies last year of 5-year summary file that is being used.
silent	Optional, defaults to TRUE. If FALSE, prints some indications of progress.
writefile	Optional, defaults to TRUE. If TRUE, saves template of needed variables as "variables needed template.csv" file to folder.

Value

Returns data.frame of info on which variables are needed from each table, much like annotated version of lookup.acs.

See Also

get.acs which uses this

---

unzip.datafiles *Unzip ACS datafile per state, downloading missing ones first*

Description

Unzip ACS datafile for each specified US State, extracting specified table(s), downloading missing zip files first.

Usage

```
unzip.datafiles(
  tables,
  mystates,
  folder = getwd(),
  end.year = acsdefaultendyearhere_func(),
  testing = FALSE,
  attempts = 5,
  silent = FALSE
)
```

Arguments

tables	Character vector of table numbers needed such as 'B01001'
mystates	Character vector of 2-character state abbreviations. Default is all states.
folder	Default is current working directory.
end.year	optional – specifies last year of 5-year summary file.
testing	Default is FALSE. If TRUE, prints more info.
attempts	Default is 5, specifies how many tries (maximum) for unzipping before trying to redownload and then give up.
silent	Default is FALSE. Whether to send progress info to standard output.

Value

Side effect is unzipping file on disk (unless testing=TRUE)

See Also

get.acs

`url.to.find.zipfile` *Get URL(s) for FTP site folder(s) with ACS 5-year summary file data*

Description

Returns URL(s) of folders (on Census Bureau FTP site) with zip file(s) based on end year.

Usage

```
url.to.find.zipfile(
  mystates,
  end.year = acsdefaultendyearhere_func(),
  url.prefix
)
```

Arguments

`mystates`	Character vector of one or more states/DC/PR, as 2-character state abbreviations. Default is all states/DC/PR.
`end.year`	Optional end year for 5-year summary file, as character, but ignored if url.prefix is specified
`url.prefix`	Optional character element that defaults to what is returned by `get.url.prefix(end.year)`

Details

See help for download.lookup.acs for more details on the URLs used for the data.
The zip files look like this for example: "20135dc0001000.zip"

The 2009-2013 summary file by state-seqfile combo is in folders that look like this:

ftp://ftp.census.gov/acs2013_5yr/summaryfile/2009-2013_ACSSF_By_State_By_Sequence_Table_Subset/DistrictOfColumbia/Tracts_Block_Groups_Only

The 2008-2012 summary file by state-seqfile combo is in folders that look like this:

http://www2.census.gov/acs2012_5yr/summaryfile/2008-2012_ACSSF_By_State_By_Sequence_Table_Subset/Alabama/Tracts_Block_Groups_Only

The 2007-2011 summary file by state-seqfile combo is in folders that look like this:

ftp://ftp.census.gov/acs2011_5yr/summaryfile/2007-2011_ACSSF_By_State_By_Sequence_Table_Subset/DistrictOfColumbia/Tracts_Block_Groups_Only/
http://www2.census.gov/acs2011_5yr/summaryfile/2007-2011_ACSSF_By_State_By_Sequence_

Table_Subset/DistrictOfColumbia/Tracts_Block_Groups_Only/
URL must be the ftp site, not the http version.

But 2010-2014 was on http only, not ftp, as of mid Dec 3 2015 release day.

Value

Returns character vector that is URL(s) such as "ftp://ftp.census.gov/acs2012_5yr/summaryfile"

See Also

get.acs, url.to.find.zipfile, download.geo

---

which.seqfiles *Find Which Sequence Files Contain Given ACS Table(s)*

Description

The US Census Bureau provides 5-year summary file data from the American Community Survey in sequence files on their FTP site. This function reports which sequence files contain the specified tables. Used by get.acs

Usage

```
which.seqfiles(tables, lookup.acs, end.year = acsdefaultendyearhere_func())
```

Arguments

tables	character vector, required. Defines which ACS table(s) to check, such as 'B01001'
lookup.acs	data.frame, optional (if not provided then it is downloaded from Census). Specifies what variables are in which tables and which tables are in which sequence files on the FTP site.
end.year	Character element, optional. Defines end year for 5-year dataset. Valid years are limited. Ignored if lookup.acs is specified, however. If they imply different years, the function stops with an error message.

Value

Returns a vector of one or more numbers stored as characters, each defining one sequence file, such as "0001".

See Also

get.acs and acs.lookup from the **acs** package, which does something related but is more flexible & robust. Also see get.acs which uses this.

`zipfile` *Get name(s) of zip file(s) for ACS 5-year summary file data*

Description

Returns name(s) of zip file(s) based on state(s), a sequence file number, a prefix, and end year.

Usage

```
zipfile(
  mystates,
  seqfilenum,
  zipfile.prefix,
  end.year = acsdefaultendyearhere_func()
)
```

Arguments

`mystates`	Required vector of 2-character state abbreviation(s)
`seqfilenum`	Required single sequence file number used by ACS 5-year summary file
`zipfile.prefix`	Optional character element, defaults to value looked up based on end.year.
`end.year`	Optional end year for 5-year summary file, as character, like '2018'

Value

Returns character element that is name of zip file such as "20115dc0113000.zip"

See Also

`get.acs`

Index

∗ **datasets**
  geo, 13
  geoformat2018, 14
  geoformat2019, 15
  geoformat2020, 16
  lookup.acs, 36
  lookup.acs2018, 38
  lookup.acs2020, 39

acs, *13*, *21*
acs.lookup, *11*, *26*, *38, 39*, *51*
acsdefaultendyearhere, 3
acsdefaultendyearhere_func, 3
ACSdownload, 3
ACSdownload-package *(*ACSdownload*)*, 3
acsfirstyearavailablehere, 4

clean.fips, *43*
clean.mystates, 5, *8*
clean.sumlevel, 6

datafile, 7, *24*, *32*, *44, 45*
download.datafiles, *5, 6*, 7
download.geo, 8, *14*, *29*, *31*, *47*, *51*
download.lookup.acs, 9, *26*, *50*

ejscreen.download, *21*

format_est_moe, 11
format_for_acs_package, 12

geo, 13
geofile, *9*, *14*, 14, *24*, *32*, *44, 45*
geoformat2018, 14
geoformat2019, 15
geoformat2020, 16
get.acs, *3–9*, *11–16*, 17, *24–27*, *29–32*, *35*, *38–41*, *44–47*, *49–52*
get.acs.all, 23
get.bg, 24, *31*
get.datafile.prefix, 24, *32*, *44, 45*
get.distances, *8*
get.field.info, *25*, 25, *26*, *29, 30*
get.lookup.acs, *11*, *25*, 26, *27*, *32*, *34*, *38, 39*, *48*
get.lookup.file.name, *11*, *26*, 27, *32*
get.read.geo, *9*, *13, 14*, 27, *35*, *46*
get.state.info, *6*
get.table.info, *25, 26*, *29*, 29, *30*
get.table.info2, *29*, 30
get.tracts, *24*, 31
get.url.prefix, 31, *50*
get.url.prefix.lookup.table, *11*, *26, 27*, 32
get.zipfile.prefix, *24*, *32*, 32, *44, 45*
getseqnumsviafilenames, 33
getstatesviafilenames, 33
gettablesviaseqnums, 34

intersperse, *11*

join.geo.to.tablist, 34

list_files_ftp, 35
list_files_ftp_worker, 36
lookup.acs, 36
lookup.acs2018, 38
lookup.acs2020, *3*, 39
lookup.states, *6*

merge_tables, 40

nhgis, *3*, *21*, 40, *42–45*
nhgisfind, 42
nhgisfips, 43
nhgisread, *41–43*, 44, *45*
nhgisreadcodebook, *44*, 45

read.acs, *12, 13*
read.concat.states, *33, 34*, 45
read.geo, 46

set.needed, *3*, *18*, *46*, 48

tidycensus, *4*, *17*, *21*

unzip.datafiles, 49
url.to.find.zipfile, *9*, *31*, 50, *51*

which.seqfiles, 51

zipfile, 52