Package 'ejscreen'

February 8, 2023

Title EJSCREEN Tools for US EPA Environmental Justice (EJ) Mapping and Screening

Description Data and tools related to the United States Environmental Protection Agency's screening and mapping tool for environmental justice, EJSCREEN. For any imported/suggested packages not on CRAN, see http://ejanalysis.github.io

Version 2.1.1

Date 2022-11-11

Imports ACSdownload,
Hmisc,
data.table,
devtools,
RCurl,
readr

Suggested proxistat

Depends ejanalysis,
analyze.stuff,
R (>= 3.5.0)

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

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

RoxygenNote 7.2.3

License MIT + file LICENSE

Repository GitHub

Author info@ejanalysis.com

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

NeedsCompilation no

LazyData true

Encoding UTF-8

R topics documented:

`add_metadata` *helper function for package to set attributes of a dataset*

Description

This can be used annually to update some datasets in a package. It just makes it easier to set a few metadata attributes similarly for a number of data elements, for example, to add new or update existing attributes.

Usage

```
add_metadata(x, metadata)
```

Arguments

x	dataset (or any object) whose metadata you want to update or create
metadata	must be a named list, so that the function can do this for each i: attr(x, which=names(metadata)[i]) <- metadata[[i]]

Value

returns x but with new or altered attributes

Examples

```
x <- data.frame(a=1:10,b=1001:1010)
metadata <- list(
census_version = 2020,
acs_version = '2016-2020',
acs_releasedate = '3/17/2022',
ejscreen_version = '2.1',
ejscreen_releasedate = 'October 2022',
ejscreen_pkg_data = 'bg22'
)
x <- add_metadata(x, metadata)
attributes(x)
x <- add_metadata(x, list(status='final'))
attr(x,'status')
```

---

bg22 *blockgroup data from the EPA EJScreen dataset (ACS 2016-2020, etc.)*

Description

```
This is the EPA EJScreen dataset version 2.1 with additional columns added.
See also the EJAMejscreendata and EJAM packages once available.

Contents

Columns are added for bg22 that are not in the EPA version on the FTP site:
- lat lon for bg centroids
- state name, state abbrev, county name, and FIPS for tract, county, state
- demographic subgroups (percent Hispanic, etc.)
 Detailed info on demog subgroups was in bg22DemographicSubgroups2016to2020
 Column renamed to friendlier variable names than the ones on the FTP site.

Vintage

- EJScreen 2.1 was released by EPA in October 2022.
  It is called bg22 here.
  It uses ACS 2016-2020, which the Census Bureau released March 17, 2022
  (delayed from Dec 2021).
```

```
- EJScreen 2.0 was released by EPA February 18, 2022.
  It was called bg21 here.
  It used ACS 2015-2019, which the Census Bureau released December, 2020.
  (EJScreen 2.0 was called the "2021 version" and bg21 here
  because it would have been called the late 2021 version but was delayed).
```

Format

data.frame with 242335 rows (block groups) and approx 175 columns

---

`bg22DemographicSubgroups2016to2020`

Demographic subgroups of race/ethnicity by block group

Description

bg22DemographicSubgroups provides subgroups that are the ones making up EJScreen's such as Hispanic or Latino, Non-Hispanic Black Alone, etc.

Details

```
 This dataset is a companion to the block group data from EJScreen.
EJScreen and therefore bg22 would have lacked table B03002 (race ethnicity)
 so that table is obtained as bg22DemographicSubgroups

This also includes race/ethnicity data for Puerto Rico, but not GU/AS/VI/MP.

 EJScreen 2.1 uses ACS2020, which is from 2016-2020
 (released March 17 2022, delayed from Dec 2021).
 It was to be called the 2022 version of EJScreen, and
 here is called bg22.

EJScreen 2.0 was released by EPA 2022-02-18 (delayed from mid/late 2021).
  EJScreen 2.0 used ACS2019, which is from 2015-2019 (released Dec 2020).
  It was to be a late 2021 version, and here had been called bg21.

bg22DemographicSubgroups was created by downloading and calculating
RACE ETHNICITY SUBGROUP VARIABLES THAT ARE NOT IN EJSCREEN
(the subgroups within "minority" or "people of color").
This is from Census Table B03002, and includes percent Hispanic,
percent Non-Hispanic Black Alone (not multirace), etc.
Race ethnicity groups are defined by Census Bureau. They are
mutually exclusive (no overlaps between groups,
so a person is always in only one of these groups)
so they add up to the total population count or percent.
Block group resolution for USA.
From Census ACS 5 year summary file.

This will give a quick look at some key stats:
```

```
  # round(data.frame(cbind(
  # subgroups=unlist(ustotals(bg22DemographicSubgroups2016to2020)),
  # maingroups = unlist(ustotals(subset(bg22, bg22$ST !='PR')))
  # ),2)

##############################################################################
How dataset was created:
##############################################################################

  # see  ejscreen/inst/SCRIPT_create_bgDemog_ejscreen2.1_andtracts.R
  # the SCRIPT for how this was created
  # and  ejscreen/inst/SCRIPT_ADD_PUERTORICO_DEMOG_SUBGROUPS.R for PR part.

  # DOWNLOADED ACS TABLE WITH RACE ETHNICITY BY BLOCK GROUP
  # AND CREATE PERCENT VARIABLES LIKE PERCENT HISPANIC, ETC.

# These are created: (count and percent hispanic or latino,
 # nonhispanic white alone i.e. single race,
# nonhispanic black or african american alone, Not Hispanic or Latino
#   American Indian and Alaska Native alone,
# Not Hispanic or Latino Native Hawaiian and Other Pacific Islander alone,
# and nh some other race alone, and nh two or more races):

# "hisp"          "nhwa"      "nhba"       "nhaiana"  "nhaa"  "nhnhpia"
# "nhotheralone"  "nhmulti"   "nonmins"    "pcthisp"  "pctnhwa" "pctnhba"
# "pctnhaiana"    "pctnhaa"   "pctnhnhpia" "pctnhotheralone" "pctnhmulti"
```

`change.fieldnames.ejscreen.csv`

Change colnames of csv file on EJSCREEN FTP site to nicer colnames

---

Description

Just a wrapper to help easily change colnames used in csv file on EJSCREEN FTP site into friendlier, preferred colnames for work in R. Uses analyze.stuff::change.fieldnames()

Usage

```
change.fieldnames.ejscreen.csv(mynames)
```

Arguments

`mynames`	A character vector of colnames from a data.frame, like names(mydf). No default.

Value

Returns a character vector of colnames, same length as input parameter

See Also

ejscreenformulas ejscreen.acs.rename analyze.stuff::change.fieldnames()

Examples

```
## Not run:
gdbtable <- ejscreen.download()
names(gdbtable) <- change.fieldnames.ejscreen.csv(names(gdbtable))

## End(Not run)
```

---

ejformula *See formula(s) used for EJSCREEN variable(s)*

Description

Just a convenient way to look at the formula(s) used to create one or more variables in EJSCREEN.

Usage

```
ejformula(fieldname = "all", decreasing = NA, dropNA = TRUE, recursive = FALSE)
```

Arguments

fieldname	Optional, character vector specifying variable(s) in ejscreenformulas$Rfieldname, default is all ejscreenformulas$Rfieldname that are not NA values.
decreasing	Optional, passed to sort except default is not sorted (just the order that exists in ejscreenformulas)
dropNA	Be careful: Optional, default is TRUE. If TRUE, returns only formulas that are not NA values. If FALSE, and decreasing is not specified (sorting drops NA values here), returns vector the same length as fieldname (unless recursive = TRUE)
recursive	Optional, default is FALSE. If TRUE, returns also returns formula(s) for variable(s) found on right hand side of formula(s), i.e. those used to create specified variable(s)

Value

Character vector of the formula(s) used to calculate the specified variable, in ejscreenformulas

See Also

ejscreenformulas

Examples

```
ejformula('VSI.eo')
ejformula(c('pctmin', 'pctlowinc'))
ejformula('VSI.eo', recursive = TRUE)
ejformula()
```

`ejscreen` *Tools for EJSCREEN, US EPA's Environmental Justice (EJ) Screening and Mapping Tool*

Description

This R package provides tools related to environmental justice (EJ) analysis, specifically related to the United States Environmental Protection Agency (EPA) screening and mapping/GIS tool called EJSCREEN. See http://www.epa.gov/ejscreen This package facilitates development of the EJSCREEN dataset, based on user-provided environmental indicators. The resulting dataset is a data.frame that contains data on demographics (e.g., percent of residents who are low-income) and user-provided local environmental indicators (e.g., an air quality index), and calculated indicators called EJ Indexes, which combine environmental and demographic indicators. The dataset also provides each key indicator as a national population-percentile that represents what percentage of the US population have equal or lower raw values for the given indicator. The dataset has one row per spatial location (e.g., Census block group).

Details

Key datasets are bg22, ejscreenformulas, and the percentile lookup tables

Key functions include

- `ejscreen.download` To download the raw data from the FTP site.
- `ejscreen.create` To create a dataset of demographic indicators, EJ indexes, etc. starting with your own environmental indicators and taking demographic raw data from the American Community Survey (ACS).
- `ejscreen.lookuptables` To create the file that shows percentiles for each indicator
- Various functions from the **ejanalysis** package are also relevant.

References

http://www.epa.gov/ejscreen
https://github.com/ejanalysis/ejscreen
http://ejanalysis.github.io
http://www.ejanalysis.com/

`ejscreen.acs.calc` *Create Calculated EJSCREEN Variables*

Description

Use specified formulas to create calculated, derived variables such as percent low income. Relies upon analyze.stuff::calc.fields() from **analyze.stuff** package.

Usage

```
ejscreen.acs.calc(
  bg,
  folder = getwd(),
  keep.old,
  keep.new,
  formulafile,
  formulas
)
```

Arguments

bg	Data.frame of raw demographic data counts, and environmental indicators, for each block group, such as population or number of Hispanics.
folder	Default is getwd(). Specifies path for where to read from (if formulafile specified) and write to.
keep.old	Vector of variables names from names(bg), indicating which to return (retain, not drop). Default is to keep only the ones that match the list of default names in this code. Or this can be simply 'all' which means keep all input fields.
keep.new	Vector of variables names of new created variables, indicating which to return (retain, not drop). Default is to keep a specific list of fields (see source code). Or this can be simply 'all' which means keep all new fields.
formulafile	Name of optional csv file with column called formula, providing R syntax formulas as character fields. If not specified, function gets this from data(ejscreenformulas). Example of one formula: 'pctunder5 <- ifelse(pop==0,0, under5/pop)' Use a result of zero in cases where the denominator is zero, to avoid division by zero. For example, the formula `'pctmin <- ifelse(pop==0,0, as.numeric(mins ) / pop)'` indicates that percent minority is calculated as the ratio of number of minorities over total population of a block group, but is set to zero if the population is zero.
formulas	Options vector of formulas as character strings that contain R statements in the form "var1 <- var2 + var3" for example. Either formulafile or formulas can be specified (or neither) but not both (error). Formulas should be in the same format as a formulafile field or the contents of ejscreenformulas (via data(ejscreenformulas) or lazy loading like x <- ejscreenformulas).

Value

Returns a data.frame with some or all of input fields (those in keep.old), plus calculated new fields (those in keep.new).

Examples

```
set.seed(99)
envirodata=data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  air=rlnorm(1000), water=rlnorm(1000)*5, stringsAsFactors=FALSE)
demogdata=data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  pop=rnorm(n=1000, mean=1400, sd=200), mins=runif(1000, 0, 800),
  num2pov=runif(1000, 0,500), stringsAsFactors=FALSE)
demogdata$povknownratio <- demogdata$pop
x=ejscreen.acs.calc(bg=demogdata)
```

`ejscreen.acs.rename` *Rename Fields of ACS Data for Use in EJSCREEN*

Description

Start with raw counts from demographic survey data, and environmental data, and rename fields to use friendly variable names.

Usage

```
ejscreen.acs.rename(acsraw, folder = getwd(), formulafile)
```

Arguments

`acsraw`	Data.frame of raw data counts for each block group, such as population or number of Hispanics.
`folder`	Default is getwd(). Specifies path for where to read from (if formulafile specified) and write to.
`formulafile`	Default if this is blank is to use data('ejscreenformulas'). Otherwise filename must be specified. If not specified, function loads this as data().

Value

Returns a data.frame with some or all of input fields, plus calculated new fields.

See Also

`ejscreenformulas` `change.fieldnames.ejscreen.csv` analyze.stuff::change.fieldnames()

`ejscreen.acsget` *Download ACS tables EJSCREEN uses, but with more race ethnicity poverty details*

Description

EJScreen 2.0 was latest as of June 2022, released early 2022, and actually used ACS2019, which is from 2015-2019 (released late 2020).

Helper function used by ejscreen.create, but can be used if one wants to obtain the more detailed relevant ACS data. The EJSCREEN-related ACS tables have more of the detailed fields than the demographic data on the EJSCREEN FTP site, because the detailed fields are used to calculate the ones retained for EJSCREEN, such as percent non-hispanic black alone, percent hispanic, percent poor (<1x poverty line) not just percent low income (<2x poverty line), etc.

Usage

```
ejscreen.acsget(
  end.year = "2020",
 tables = c("B01001", "B03002", "B15002", "C16002", "C17002", "B25034", "B23025"),
  base.path = getwd(),
  data.path = file.path(base.path, "acsdata"),
  output.path = file.path(base.path, "acsoutput"),
  vars = "all",
  sumlevel = "bg",
  write.files = TRUE,
  ...
)
```

Arguments

end.year	optional character year like 2020 specifying last of 5 years of ACS summary file
tables	Default is the ones needed for EJSCREEN - character vector list of Census data tables like B01001
base.path	optional, default is working directory; folder in which data.path and output.path subfolders are or will be created
data.path	see ACSdownload::get.acs()
output.path	see ACSdownload::get.acs()
vars	Default here is 'all' vars which is more than what ejscreen.create keeps. (or can be a vector of things like 'B01001')
sumlevel	Default here is just bg but see ACSdownload::get.acs()
write.files	Default here is TRUE but see ACSdownload::get.acs()
...	passed to ACSdownload::get.acs()

Details

Tables can include the EJScreen 2.1 tables: mytables <- c("B01001","B03002","B15002",'B23025',"B25034","C16002"
myurl <- paste0('https://data.census.gov/cedsci/table?q=acs paste(mytables, collapse= ' browseURL(myurl)
# [https://data.census.gov/cedsci/table?q=acs

```
ACSdownload::get.field.info(mytables, table.info.only = TRUE)[ , 1:2]
         ID                                                                title
 # 1 B01001                                                          SEX BY AGE
 # 2 B03002                                       HISPANIC OR LATINO ORIGIN BY RACE
 # 3 B15002 SEX BY EDUCATIONAL ATTAINMENT FOR POPULATION 25 YEARS AND OVER
 # 4 B23025           EMPLOYMENT STATUS FOR THE POPULATION 16 YEARS AND OVER
 # 5 B25034                                                 YEAR STRUCTURE BUILT
 # 6 C16002  HOUSEHOLD LANGUAGE BY HOUSEHOLD LIMITED ENGLISH SPEAKING STATUS
 # 7 C17002           RATIO OF INCOME TO POVERTY LEVEL IN THE PAST 12 MONTHS
```

C16002 replaced B16004 that was older ACS source for what had been called linguistic isolation,
now called limited English speaking households. Details on language spoken: [www2.census.gov/topics/language-use/acs/acs_tabulations-language-list.pdf] [https://data.census.gov/cedsci/table?q=acs

Value

list of data.frames, default is just block group not tracts, unlike results of ACSdownload::get.acs()

Examples

```
mytables <- c("B01001", "B03002", "B15002", 'B23025', "B25034", "C16002", "C17002")
myurl <- paste0('https://data.census.gov/cedsci/table?q=acs%20', paste(mytables, collapse= '%20'), '&y=2020'
# browseURL(myurl)
```

`ejscreen.create` *Replicate or Create EJSCREEN-like dataset from your own Environmental and Demographic Data*

Description

NOTE : does not include demographic subgroups yet see names.d.subgroups

The EJSCREEN dataset each year is already available as a dataset in this package (as bg21 for example). But if you want to recreate that kind of dataset from your own environmental data (and demographic data), this function does that. This function ejscreen.create() starts with raw environmental indicator data, and/or ACS demographics, and will create (or replicate) a full EJSCREEN dataset. The ACS demographics, if not provided to the function, are downloaded from Census and calculated by this function. The ACS 5-year summary file is used to obtain block group estimates of population count, minorities, low-income, etc. The source code for this function also contains further comments with an outline of steps involved. It relies on ejscreen.acsget() which relies on ACSdownload::get.acs(), and uses ejanalysis::addFIPScomponents() to add FIPS.TRACT, countyname, etc., and also uses ejanalysis::make.bin.pctile.cols, ejanalysis::ej.indexes(), ejanalysis::flagged(), etc. @details **Note that if non-default fieldnames are used in e and/or acsraw, those must be specified in parameters including demogvarname0, wtsvarname, keep.old (and could be reflected in prefix and suffix params as well).

This function does not create lookup tables that are used in EJSCREEN to convert a raw score in a buffer to a US,Region,or state percentile. Use ejscreen.lookuptables() to create those lookup tables of 100 population weighted percentiles and mean, for US and each EPA Region and each State; for each raw score.

Usage

```
ejscreen.create(
  e,
  acsraw,
  folder = getwd(),
  keep.old,
  formulas,
  mystates = "all",
  popup = FALSE,
  write.lookup.us = FALSE,
  write.lookup.regions = FALSE,
  write.lookup.states = FALSE,
  demogvarname0 = "VSI.eo",
  wtsvarname = "pop",
```

```
  checkfips = TRUE,
  EJprefix0 = "EJ.DISPARITY",
  EJprefix1 = NULL,
  EJprefix2 = NULL,
  ejformulasfromcode = FALSE,
  ejtype = 1,
  demogvarname0suffix = "eo",
  end.year,
  threshold = FALSE,
  cutoff = 0.8,
  thresholdfieldnames,
  ...
)
```

Arguments

e	Data.frame of raw data for environmental indicators, one row per block group, one column per indicator.
acsraw	Optional data.frame of raw demographic indicators. Downloaded if not provided as parameter.
folder	Optional, default is getwd(). Passed to ACSdownload::get.acs() if demog data must be downloaded. Passed to but not currently used by ejscreen.acs.rename which uses analyze.stuff::change.fieldnames() in **analyze.stuff** package. Not currently passed to ejscreen.acs.calc which uses analyze.stuff::calc.fields() in **analyze.stuff** package.
keep.old	optional vector of colnames from e that are to be used/returned. For nondefault colnames, this must be used.
formulas	optional, see ejscreen.acs.calc for details. Defaults are in ejscreenformulas$formula Note that if formulas is specified, ejformulasfromcode is ignored.
mystates	optional vector of 2-letter state abbreviations. Default is "all" which specifies all states plus DC (BUT NOT PR - we exclude PR so that calculating US percentiles works right)
popup	optional, default is FALSE, whether to add columns of text that can be used for popup info on maps, with raw value and percentile and units for each envt, demog, and EJ indicator (US Percentiles only?)
write.lookup.us	whether to save file with lookup table of US percentiles and means
write.lookup.regions	whether to save file with lookup table of REGION percentiles and means
write.lookup.states	whether to save file with lookup table of State percentiles and means
demogvarname0	optional, default is 'VSI.eo' used as demographic indicator for EJ Indexes. Must be a colname in acsraw or created and kept by formulas.
wtsvarname	optional, default is 'pop' used for weighted percentiles, etc. Must be a colname in acsraw or created and kept by formulas.
checkfips	optional, default is TRUE. If TRUE, function checks to verify all FIPS codes appear to be valid US FIPS (correct number of characters, adding any leading zero needed, and checking the first five to ensure valid county). To use something other than actual US FIPS codes, set this to FALSE.

`EJprefix0` optional, default is 'EJ.DISPARITY' - specifies prefix for colnames of main EJ Indexes, with a period separating prefix from body of colname

`EJprefix1` optional, default is NULL, none used (old way was 'EJ.BURDEN' - specifies prefix for colnames of Alternative 1 version of EJ Indexes, with a period separating prefix from body of colname)

`EJprefix2` optional, default is NULL, none used (old way was 'EJ.PCT' - specifies prefix for colnames of Alternative 2 version of EJ Indexes, with a period separating prefix from body of colname)

`ejformulasfromcode` optional, default is FALSE. If TRUE, use EJ Index formulas built into this function instead of the EJ Index formulas in ejscreenformulas. The parameters such as demogvarname0 are only used if ejformulasfromcode=TRUE. Note that if formulas is specified, ejformulasfromcode is ignored.

`ejtype` optional, default is 1, defines which formula to use for ejindex if not using ejscreenformulas. See ejanalysis::ej.indexes But note alt1 and alt2 still use type 5 and 6 ignoring ejtype.

`demogvarname0suffix` optional, default is 'eo' - specifies suffix for colnames of EJ Indexes based on demogvarname0, with a period separating body of colname from suffix

`end.year` optional to pass to ACSdownload::get.acs() such as end.year='2013' – otherwise uses default year used by ACSdownload::get.acs()

`threshold` optional, default is FALSE. Set to TRUE to add a column (called 'flagged') to results that is TRUE when one or more of certain percentiles (US EJ Index) in a block group (row) exceed cutoff. A field called flagged can also be added via ejanalysis::flagged() ejanalysis::flagged() or via ejscreen.download(addflag = TRUE)

`cutoff` optional, default is 0.80 (80th percentile). If threshold=TRUE, then cutoff defines the threshold against which percentiles are compared.

`thresholdfieldnames` optional, default is standard EJSCREEN EJ Indexes built into code. Otherwise, vector of character class fieldnames, specifying which fields to compare to cutoff if threshold=TRUE.

`...` optional extra parameters passed only to ACSdownload::get.acs such as new.geo = FALSE, save.files = TRUE, write.files = TRUE

Value

Returns a data.frame with full ejscreen dataset of environmental and demographics indicators, and EJ Indexes, as raw values, US percentiles, but not actually the text for map popups. Output has one row per block group.

See Also

`ejscreen.lookuptables`

Examples

```
## Not run:
set.seed(99)
envirodata=data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  air=rlnorm(1000), water=rlnorm(1000)*5, stringsAsFactors=FALSE)
```

```
demogdata=data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  pop=rnorm(n=1000, mean=1400, sd=200), mins=runif(1000, 0, 800),
  num2pov=runif(1000, 0,500), stringsAsFactors=FALSE)
demogdata$povknownratio <- demogdata$pop
# downloads ACS demographics and combines with user provided envirodata:
# bg1=ejscreen.create(envirodata, mystates=c('de','dc'))
# currently does not work for nonstandard colnames
#  unless keep.old used as follows (work in progress):
y=ejscreen.create(e=envirodata, acsraw=demogdata,
  keep.old = c(names(envirodata), names(demogdata)),
  demogvarname0 = 'pctmin',wtsvarname = 'pop' )


## End(Not run)
```

`ejscreen.download` *Download the EJSCREEN Dataset csv for use in R and rename variables*

Description

Download EJSCREEN dataset from FTP site, unzip if necessary, import to R as data.table, renaming fields with friendly colnames, optionally adding a flag field (see parameter called `addflag`). Note that since 2020v, State percentiles are also available in a separate zipped csv.

Usage

```
ejscreen.download(
  folder = getwd(),
  yr = NULL,
  ftpurlbase = "https://gaftp.epa.gov/EJSCREEN/",
  justreadname = NULL,
  statepctiles = FALSE,
  addflag = FALSE,
  cutoff = 80,
  or.tied = TRUE
)
```

Arguments

`folder`	Optional path to folder (directory) where the file will be downloaded and unzipped. Default is current working directory.
`yr`	Default is latest available year found as a folder on the FTP site. Optional numeric year designating EJSCREEN version such as 2015, 2016, 2017, 2018, 2019, 2020, etc.
`ftpurlbase`	Optional. where to find the zipped data.
`justreadname`	Optional character file name - if specified, skips downloading and just tries to read previously-downloaded csv found in `folder`.
`statepctiles`	Optional, default FALSE. If TRUE, gets State Percentiles csv instead of the USPR file.

addflag	Optional. Default is FALSE. If TRUE, it adds a field called flagged, which is TRUE if 1 or more of the EJ Indexes is at/above the cutoff US percentile.
cutoff	Optional. Default is 80. See addflag parameter.
or.tied	Optional. Default is TRUE, meaning at or above the cutoff. FALSE means above only. See addflag parameter.

Details

Not fully tested.

```
  Each version of EJSCREEN uses updated environmental data and updated 5-year
  summary file estimates from the American Community Survey (ACS).

  The 2017-2021 American Community Survey 5-year estimates are scheduled to
  be released on Thursday, December 8, 2022.
  The 2016-2020 ACS 5-year estimates were released Dec 2021.

  EJScreen 2.0 (released February 2022),
    actually uses ACS2019 5-year summary file data, which is from 2015-2019
     (released Dec 2020).
    It is avail as data in \link{bg21}
  (EJScreen 2.0 would have been called the 2021 version in old naming scheme).

 Note the 2020 version of EJSCREEN
 (confusingly released early/mid 2021 not late 2020)
    actually used ACS2018, which is from 2014-2018 (released late 2019).
    It had been called bg20

       TO JUST READ THE EJSCREEN DATA ONCE DOWNLOADED FROM THE FTP SITE:

      May recode ejscreen.download to split out just the part that
       downloads, unzips, reads into R verbatim.

 x <- readr::read_csv('~/Downloads/EJSCREEN_2020_USPR.csv', na = 'None')
 x <- data.frame(x, stringsAsFactors=FALSE) #if want data.frame not data.table

       TO JUST RENAME COLUMNS TO FRIENDLY NAMES USED IN THIS PACKAGE:

       names(x) <- ejscreen::change.fieldnames.ejscreen.csv(names(x))

       TO JUST ADD SOME USEFUL COLUMNS (FIPS, countyname, statename, etc.):

       x <- ejanalysis::addFIPScomponents(x)
       # and could add attribute to indicate vintage
```

Value

Returns a data.frame with ejscreen dataset of environmental and demographics indicators, and EJ Indexes, as raw values, US percentiles, text for popups. Output has one row per block group, sorted by FIPS.

Source

See http://www.epa.gov/ejscreen for more information, and see http://www.epa.gov/ejscreen/download-ejscreen-data

See Also

ejscreen.create change.fieldnames.ejscreen.csv

Examples

```
# bg <- ejscreen.download('~')
## bg <- ejscreen.download('~',
#  justreadname = 'EJSCREEN_Full_USPR_2020.csv')
# bg <- bg[ , !grepl(pattern = 'pctile\.text', x = names(bg))]
# bg <- bg[ , !grepl(pattern = 'svi6', x = names(bg))]
# setwd('~')
# save(bg, file = 'bgYEAR20XX.rdata')
```

---

ejscreen.lookuptables *Create EJSCREEN Lookup Tables of Pop. Percentiles by Zone - but see ejscreen::write.wtd.pctiles.by.zone()*

Description

*** Was work in progress as of 2022 ...maybe this has more useful code in some parts
see also:

ejanalysis::write.wtd.pctiles.by.zone

ejscreen::ejscreen.lookuptables
ejscreen/inst/SCRIPT_pctilelookups_Create new lookup tables for EJScreen 2.0.R
ejscreen/inst/SCRIPT_pctilelookups_read-downloaded-pctile-lookups.R
will recode to use
ejanalysis::write.wtd.pctiles.by.zone
and see
table.pop.pctile and
map service with lookup tables
or check gdb for lookup tables?

Start with raw environmental, demographic, and EJ indicator data, and write as csv files to disk a series of lookup tables that show population percentiles and mean values for each indicator.

Usage

```
ejscreen.lookuptables(
  x,
  weights,
```

```
  cols,
  zonecols = c("ST", "REGION"),
  zoneOverallName = "USA",
  folder = getwd(),
  filename1 = "lookupUSA",
  filenameprefix = "lookup",
  savefilezoneset = TRUE,
  savefileperzone = FALSE,
  missingcode = NA
)
```

Arguments

x	Data.frame of indicators, one row per block group, one column per indicator.
weights	Weights for percentiles – Default is x$pop (if found in x, otherwise no weights) (population count to provide population percentiles.)
cols	Optional vector of colnames of x that need percentile lookup tables, or `all` which means all numeric fields in x. Default is a standard set of EJSCREEN fieldnames defined within this function (see source code).
zonecols	Optional. Must set to NULL if no zones wanted, because default is `c('ST', 'REGION')`, names of cols in x that contain zone codes, such as State names or Region numbers, used to create a lookup table file for each of the zonecols, with separate percentiles calculated within each zone.
zoneOverallName	Name of entire domain to use in table column called REGION. Default is USA.
folder	Default is `getwd()` - specifies where to save the csv files.
filename1	specifies name of file saved, but .csv is added after this
filenameprefix	specifies first part of names of files saved for zones
savefilezoneset	Save a file for the entire set of zones of one type, like 1 file containing stats on all states
savefileperzone	Save one file for each zone, like each state
missingcode	Leave this unspecified if missing values are set to NA in the input data. Default is -9999999 (but if already NA then do not specify anything for this). The number or value in the input data that designates a missing value.

Details

As of mid-2022 EJScreen was changing from pop weighted to unweighted percentiles, and changing so low tied values are 0th and percentile will mean percent <x not percent <= x

Percentiles are calculated as exact values and then rounded down to the nearest 0-100 percentile. This calculates percentiles among only the non-NA values. In other words, people in places with missing data are excluded from the calculation. This means the percentile is the percent of people with valid data (i.e., not NA) who have a tied or lower value.

Value

Overall lookup table(s) as data.frame (but not zonal ones). Creates lookup tables saved as csv files to specified folder. One table for overall percentiles, and one for each of the zonecols (unless that is set to NULL).

Examples

```
## Not run:
# ejscreen dataset:
bg <- bg22
out <- ejscreen.lookuptables(ejscreen::bg[bg$REGION %in% 2:3,], weights=bg$pop[bg$REGION %in% 2:3,])
# Try with a sample envt data set:
set.seed(99)
envirodata <- data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  pm = runif(1000,5,20), o3 = runif(1000,3,50),
  air=rlnorm(1000), water=rlnorm(1000)*5, stringsAsFactors=FALSE)
demogdata <- data.frame(FIPS=analyze.stuff::lead.zeroes(1:1000, 12),
  pop = rlnorm(1000, meanlog = log(1000), sdlog = 1), stringsAsFactors=FALSE)
x <- ejscreen.lookuptables(envirodata, weights=demogdata$pop, cols='all', zonecols=NULL)
x

## End(Not run)
```

---

ejscreen.rollup *Aggregate EJSCREEN Dataset at Lower Resolution (e.g., Tracts)*

Description

Start with full EJSCREEN dataset at one resolution (typically block groups), and create aggregated data at a higher geographic scale (e.g., tracts or counties)

Usage

```
ejscreen.rollup(
  bg,
  fipsname = "FIPS.TRACT",
  scalename = "tracts",
  enames,
  folder = getwd(),
  sumnames,
  avgnames,
  wts,
  acsnames,
  checkfips,
  ...
)
```

Arguments

bg	Data.frame of raw data for environmental and demographic counts, one row per block group typically, one column per indicator.
fipsname	Default is 'FIPS.TRACT' - specifies colname of unique ID field FIPS used to group by. Can be FIPS.TRACT, FIPS.COUNTY, FIPS.ST, or REGION in default dataset.
scalename	***Not used. Default is 'tracts' - specifies text to use in naming the saved file.
enames	Default is names.e, the colnames of raw envt indicators in bg
folder	***Not used. Optional, default is getwd().

sumnames	Default is a vector of colnames in bg, those which should be rolled up as sums with na.rm=TRUE (e.g., sum of all block group population counts in the tract) including 'pop', 'povknownratio', 'age25up', 'hhlds', 'builtunits', 'mins', 'lowinc', 'lths', 'lingiso', 'under5', 'over64', 'VNI.eo', 'VNI.svi6', 'VDI.eo', 'VDI.svi6', 'hisp', 'nhaa', 'nhaiana', 'nhba', 'nhmulti', 'nhnhpia', 'nhotheralone', 'nhwa', 'nonmins', 'area', 'pre1960'
avgnames	Default is ejscreen::enames, a vector of colnames in bg, those which should be rolled up as weighted averages (e.g., pop wtd mean of air pollution level)
wts	Default is 'pop', the colname in bg specifying the field to use when calculating the weighted mean of all blockgroups in a tract, for example.
acsnames	Not used. Default is a vector of demographic colnames in bg, used in default ejscreen dataset (see code or ejscreenformulas)
checkfips	Whether to try to validate FIPS, passed to ejscreen.create(). See source for default.
...	Optional parameters to pass to ejscreen.create which uses formulas to create indicators from raw values.

Details

**default fieldnames are assumed for now. Uses ejscreen.create

Value

Returns a data.frame with ejscreen dataset of environmental and demographics indicators, and EJ Indexes, as raw values, US percentiles, but not text for popups. *** Output has one row per tract, county, state, or region, depending on what is specified.

See Also

ejscreen.create

Examples

```
## Not run:
 # load("~/Dropbox/EJSCREEN/R analysis/bg 2015-04-22 Rnames plus subgroups.RData")
 # Do this for each of several levels of resolution
 #
 fipsnames <- c('FIPS.TRACT', 'FIPS.COUNTY', 'FIPS.ST', 'REGION')
 scalenames <- c('tracts', 'counties', 'states', 'regions')
 # or just for tracts, say this:
 #   fipsnames <- 'FIPS.TRACT'; scalenames <- 'tracts'

 for (i in 1:length(fipsnames)) {

 ################################### #
 # Specify resolution of interest
 fipsname <- fipsnames[i] # 'FIPS.TRACT'
 scalename <- scalenames[i] # 'tracts'

 ################################### #
 # Get results, using the function
 myrollup <- ejscreen.rollup(bg = bg, fipsname = fipsname)

 ################################### #
```

```
  # Save results
  save(myrollup, file = paste('EJSCREEN 202o', scalename, 'data.RData') )
 write.csv(myrollup, row.names = FALSE, file = paste('EJSCREEN 2020', scalename, 'data.csv'))

  }

## End(Not run)
```

`ejscreen.rollup.all` *Aggregate EJSCREEN Dataset at Lower Resolutions (e.g., Tracts and Counties)*

Description

Does what ejscreen.rollup does, but for more than one resolution - a batch of rollups done at once. Start with full EJSCREEN dataset at one resolution (typically block groups), and create aggregated data at higher geographic scales (e.g., tracts and counties)

Usage

```
ejscreen.rollup.all(
  bg,
  scalenames = c("tracts", "counties", "states", "regions"),
  fipsnames = c("FIPS.TRACT", "FIPS.COUNTY", "FIPS.ST", "REGION"),
  myfolder = getwd(),
  filenamebase = "EJSCREEN",
  filenames.R,
  filenames.csv,
  save.R = FALSE,
  save.csv = FALSE,
  assigning = FALSE,
  ...
)
```

Arguments

`bg`	Required, data.frame of raw data for environmental and demographic counts, one row per block group typically, one column per indicator.
`scalenames`	optional character vector of terms used to create filenames if saving files, default = c('tracts', 'counties', 'states', 'regions')
`fipsnames`	optional character vector of certain colnames in bg, used to select columns from bg to summarize by, default = c('FIPS.TRACT', 'FIPS.COUNTY', 'FIPS.ST', 'REGION'),
`myfolder`	optional folder path for saving files, default = getwd()
`filenamebase`	optional character element, default = 'EJSCREEN', used to construct filenames to save files if relevant.
`filenames.R`	optional vector of filenames, default has the word EJSCREEN and scalename .RData

filenames.csv	optional vector of filenames, default has the word EJSCREEN and scalename .csv
save.R	optional logical, default = FALSE, whether to save files as .RData
save.csv	optional logical, default = FALSE, whether to save files as .csv
assigning	optional logical, default = FALSE, whether to assign results to variable in calling environment, or just return list of data.frames as result.
...	Optional parameters to pass to ejscreen.create which uses formulas to create indicators from raw values. e.g. can pass clean=FALSE that is in turn passed eventually to clean.fips()

Details

**default fieldnames are assumed for now. Uses ejscreen.create.

Value

Returns a list of data.frames each like output of ejscreen.rollup, one per resolution (e.g., one for counties)

See Also

ejscreen.rollup

Examples

```
# (none)
```

---

ejscreen.rollup.save *Helper for ejscreen.rollup.all, to save files of results*

Description

Just saves csv and/or RData file(s)

Usage

```
ejscreen.rollup.save(
  myrollup,
  myfolder = getwd(),
  filenamebase = "EJSCREEN",
  scalename = c("tracts"),
  filename.R,
  filename.csv,
  save.R = TRUE,
  save.csv = TRUE
)
```

Arguments

myrollup	Required, data.frame results from ejscreen.rollup.all (just one scale at a time though)
myfolder	optional folder path for saving files, default = getwd()
filenamebase	optional character element, default = 'EJSCREEN', used to construct filenames to save files if relevant.
scalename	optional character term used to create filenames, default = c('tracts')
filename.R	optional filename, default has the word EJSCREEN and scalename .RData
filename.csv	optional filename, default has the word EJSCREEN and scalename .csv
save.R	optional logical, default = FALSE, whether to save files as .RData
save.csv	optional logical, default = FALSE, whether to save files as .csv

Value

Returns a 2 element vector with full paths of saved R and csv files (or NA instead of a path, if one of those is not saved)

See Also

ejscreen.rollup.all

---

ejscreenformulas *EJSCREEN Formulas and Fieldnames*

Description

This provides fieldnames and formulas required by the **ejscreen** package. Formulas can be viewed this way: sort(ejscreenformulas$formula)

Also useful to try using ejformula() which displays info from ejscreenformulas. [They are not in the latest version of ejscreen but still in ejscreenformulas I think] To exclude obsolete svi6-related and obsolete alternative forms of EJ Index: # substr(grep('svi6|BURDEN|PCT', x=ejformula(), value = TRUE, invert = TRUE), 1, 80)

Usage

```
data('ejscreenformulas')
```

Format

A data.frame:

> str(ejscreenformulas)
'data.frame': approx 470 obs. of 8 variables:

- $ gdbfieldname : chr NA NA NA NA ...
- $ Rfieldname : chr "ageunder5m" "age5to9m" "age10to14m" "age15to17m" ...
- $ acsfieldname : chr "B01001.003" "B01001.004" "B01001.005" "B01001.006" ...

- $ type : chr "ACS" "ACS" "ACS" "ACS" ...
- $ glossaryfieldname: chr NA NA NA NA ...
- $ formula : chr NA NA NA NA ...
- $ acsfieldnamelong : chr "Under 5 years|SEX BY AGE" "5 to 9 years|SEX BY AGE" "10 to 14 years|SEX BY AGE" "15 to 17 years|SEX BY AGE" ...
- $ universe : chr "Universe: Total population" "Universe: Total population" "Universe: Total population" "Universe: Total population" ...

Source

See related Technical Documentation at http://www.epa.gov/ejscreen

See Also

ejformula ejscreenformulasnoej names.e names.d names.ej

`ejscreenformulasnoej` *EJSCREEN Formulas and Fieldnames Excluding EJ Index Formulas*

Description

This provides fieldnames and formulas required by the **ejscreen** package. Formulas can be viewed this way: sort(ejscreen::ejscreenformulas$formula) This excludes the EJ Index formulas for cases where those are to be calculated using code separately.

Usage

```
data('ejscreenformulasnoej')
```

Format

A data.frame:

> str(ejscreenformulas)
'data.frame': 470 obs. of 8 variables:

- $ gdbfieldname : chr NA NA NA NA ...
- $ Rfieldname : chr "ageunder5m" "age5to9m" "age10to14m" "age15to17m" ...
- $ acsfieldname : chr "B01001.003" "B01001.004" "B01001.005" "B01001.006" ...
- $ type : chr "ACS" "ACS" "ACS" "ACS" ...
- $ glossaryfieldname: chr NA NA NA NA ...
- $ formula : chr NA NA NA NA ...
- $ acsfieldnamelong : chr "Under 5 years|SEX BY AGE" "5 to 9 years|SEX BY AGE" "10 to 14 years|SEX BY AGE" "15 to 17 years|SEX BY AGE" ...
- $ universe : chr "Universe: Total population" "Universe: Total population" "Universe: Total population" "Universe: Total population" ...

Source

See related Technical Documentation at http://www.epa.gov/ejscreen

See Also

ejformula ejscreenformulas names.e names.d names.ej

---

ejscreensignifarray *Specify Significant Digits for Each Column of EJSCREEN Indicators*

Description

Given a matrix or numeric data.frame, round each column to a specified column-specific number of significant digits. This function provides default values significant digits to use for an EJSCREEN environmental dataset. This is a wrapper for analyze.stuff::signifarray which is a wrapper that applies signif() to a matrix or data.frame.

Usage

```
ejscreensignifarray(dat, digits = "ejscreen")
```

Arguments

dat	Required, matrix or numeric data.frame with the values to be rounded.
digits	Optional, 'ejscreen' by default. Can be a vector as long as the number of columns in dat, where each elements specifies the number of significant digits to retain for numbers in the corresponding column of dat. If 'ejscreen' it specifies using the default settings described below in details, in which case all colnames(dat) must be among (but in any order) defaultcolnames below.

Details

Sig figs used if digits specified as 'ejscreen' are those stored in data(esigfigs)

Value

Returns dat, but with numbers rounded based on digits parameter.

See Also

esigfigs analyze.stuff::signifarray() signif

esigfigs *How many signif digits to show*

Description

How many sig figs to show in showing environmental indicators in EJSCREEN?

Usage

```
data('esigfigs')
```

Format

A data.frame:

> str(esigfigs)
'data.frame': 12 obs. of 2 variables:
$ sigfigs: num 3 3 2 2 2 3 2 2 2 2 ...
$ evar : chr "pm" "o3" "cancer" "neuro" ...

sigfigs evar
3 pm
3 o3
2 cancer
2 neuro
2 resp
3 dpm
2 pctpre1960
2 traffic.score
2 proximity.npl
2 proximity.rmp
2 proximity.tsdf
2 proximity.npdes
2 ust

Source

See related Technical Documentation at http://www.epa.gov/ejscreen

See Also

make.popup.e

ileile_plot *Scatter plot of median of Environmental score percentiles for each Demographic score percentile Does Environmental score tend to be higher where Demographic percent is higher?*

Description

Scatter plot of median of Environmental score percentiles for each Demographic score percentile Does Environmental score tend to be higher where Demographic percent is higher?

Usage

```
ileile_plot(
  x_demog_percentile,
  y_envt_percentile,
  xlab = "Demographic percentile (binned by rounding)",
  ylab = "median of Environmental percentiles",
 main = "Does Environmental indicator tend to be higher where Demographic % is higher?",
  ...
)
```

Arguments

x_demog_percentile	demographic scores of block groups, as percentiles 0-100
y_envt_percentile	environmental indicator scores of block groups, as percentiles 0-100
xlab	optional x axis text for demographic percentile variable
ylab	optional y axis text for environmental percentile variable
main	optional main title for plot
...	passed to plot(), such as col='red'

Examples

```
bg <- bg22
ileile_plot(
  ejscreen::bg$pctile.pctmin,
  ejscreen::bg$pctile.proximity.tsdf,
  xlab='low income percentile',
  ylab='median pctile of environmental score at that level of % Demographics'
 )
 # also see
 # cars::qqPlot( something_like_pctile.proximity.rmp_at_some_sites, main="How close to US distribution of sco

 pj(1,3)
```

lookupRegions *EPA-Region-level EJScreen percentile lookup tables*

Description

```
WARNING: **** As of 2022-07, EJScreen did not include these lookup tables
 for download as csv files on FTP site.

The nationwide most recent version of the EJSCREEN percentile lookup table.
 Lookup table with one column per indicator; rows 0-100 show percentiles,
 and last two rows show mean and standard deviation.

 Note the 2021 version of EJSCREEN (to be released late 2021 ??)
   actually uses ACS2019, which is from 2015-2019 (released late 2020).
 Note the 2020 version of EJSCREEN (confusingly released mid 2021)
   actually uses ACS2018, which is from 2014-2018 (released late 2019).
 Note the 2019 version of EJSCREEN (released late 2019)
   actually uses ACS2017, which is from 2013-2017 (released late 2018).
 Note the 2018 version of EJSCREEN (released late 2018)
   actually uses ACS2016, which is from 2012-2016 (released late 2017).

 This is from the EJSCREEN dataset from the ftp site but with
 fields renamed for easier use in the ejscreen package.
 It can be used with for example
 ejanalysis::lookup.pctile(13, varname.in.lookup.table = 'pm',
 lookup = lookupUSA)
 It shows what the cutpoints are for each variable at percentiles
 0,1,2 through 99, 100.
 For example, if the traffic.score is 1000 in a given location,
 you can look where that falls in the percentiles and see that 81
 the US population had lower scores:
 ejanalysis::lookup.pctile(1000, varname.in.lookup.table = 'traffic.score',
  lookup = lookupUSA)
```

Details

See lookupUSA for details.

See Also

lookupUSA lookupRegions lookupStates ejanalysis::lookup.pctile()

Examples

```
ejanalysis::lookup.pctile(1000,
varname.in.lookup.table = 'traffic.score',  lookup = lookupUSA)
    ejanalysis::lookup.pctile(c(1000, 3000),
    varname.in.lookup.table = 'traffic.score',
      lookup = lookupStates, zone = 'NY')
      # Those traffic scores are at the 62d and 83d percentiles
```

```
    within NY State (83 percent
    # of the NY State population had a traffic score lower than 3000).
  ## Not run:

   bg <- bg20[ sample(1:NROW(bg20), 100), ]

   state.pctile.pm <- ejanalysis::lookup.pctile(myvector = bg$pm,
   varname.in.lookup.table = 'pm',
      lookup = lookupStates, zone = bg$ST)
   plot(state.pctile.pm, bg$pctile.pm, pch = '.')
   text(state.pctile.pm, bg$pctile.pm,
   labels = paste(bg$ST, round(bg$pm,1)), cex = 0.8)
   abline(0,1)
   lookupStates[lookupStates$PCTILE == 'mean', c('REGION', 'pm')]
   lookupUSA[lookupUSA$PCTILE == 'mean', c('REGION', 'pm')]

## End(Not run)
```

`lookupStates` *The State-level latest version of the EJSCREEN percentile lookup table.*

Description

For vintage, see attributes(lookupStates)

```
As of 2022-07, EJScreen did not include these lookup tables for download as
 csv files on FTP site.

The nationwide most recent version of the EJSCREEN percentile lookup table.
 Lookup table with one column per indicator and rows 0-100 show percentiles,
 and last two rows show mean and standard deviation.

EJScreen 2.1 was released circa August 2022.
  EJScreen 2.1 uses ACS2020, which is from 2016-2020 (released March 17
   2022, delayed from Dec 2021).
  It was to be called the 2022 version of EJScreen, and
  here is called bg22.

EJScreen 2.0 was released by EPA 2022-02-18 (delayed from mid/late 2021).
  EJScreen 2.0 used ACS2019, which is from 2015-2019 (released Dec 2020).
  It was to be called the 2021 version, and here is called bg21 as
  it was to be a late 2021 version.

 This is from the EJSCREEN dataset from the ftp site but with
 fields renamed for easier use in the ejscreen package.
 Versions without renamed columns were USA_2022_LOOKUP and
 States_2022_LOOKUP.

 It can be used with for example ejanalysis::lookup.pctile(
 13, varname.in.lookup.table = 'pm', lookup = lookupUSA)
```

```
It shows what the cutpoints are for each variable at percentiles
0,1,2 through 99, 100.
For example, if the traffic.score is 1000 in a given location,
you can look where that falls in the percentiles and see that say 81
of the US population had lower scores:
ejanalysis::lookup.pctile(1000, varname.in.lookup.table = 'traffic.score',
  lookup = lookupUSA)
```

Details

See lookupUSA for details.

See Also

lookupUSA lookupRegions lookupStates ejanalysis::lookup.pctile()

Examples

```
ejanalysis::lookup.pctile(1000,
varname.in.lookup.table = 'traffic.score',  lookup = lookupUSA)
    ejanalysis::lookup.pctile(c(1000, 3000),
    varname.in.lookup.table = 'traffic.score',
      lookup = lookupStates, zone = 'NY')
      # Those traffic scores had been at the 62d and 83d percentiles
      within NY State (83 percent
      # of the NY State population had a traffic score lower than 3000).
   ## Not run:
    bg <- bg22[sample(1:NROW(bg22), 100), ]
    state.pctile.pm <- ejanalysis::lookup.pctile(myvector = bg$pm,
    varname.in.lookup.table = 'pm',
       lookup = lookupStates, zone = bg$ST)
    plot(state.pctile.pm, bg$pctile.pm, pch = '.')
    text(state.pctile.pm, bg$pctile.pm,
    labels = paste(bg$ST, round(bg$pm,1)), cex = 0.8)
    abline(0,1)
    lookupStates[lookupStates$PCTILE == 'mean', c('REGION', 'pm')]
    lookupUSA[lookupUSA$PCTILE == 'mean', c('REGION', 'pm')]

## End(Not run)
```

---

lookupUSA

The nationwide most recent version of the EJSCREEN percentile lookup table.

Description

For vintage, see attributes(lookupUSA)

As of 2022-07, EJScreen did not include these lookup tables for download as csv files on FTP site.

The nationwide most recent version of the EJSCREEN percentile lookup table.

```
 Lookup table with one column per indicator and rows 0-100 show percentiles,
 and last two rows show mean and standard deviation.

 The lookup table is for the US including PR.
and bg22 has PR, so the lookup table means should match means calculated from bg22.
    round(unlist(cbind(ustotals(bg22[bg22$ST != 'PR',]))['PCTMIN.US',]),5)

    round(lookupUSA[lookupUSA$PCTILE == 'mean','pctmin'],5)

EJScreen 2.1 was released circa August 2022.
 EJScreen 2.1 uses ACS2020, which is from 2016-2020 (released March 17 2022, delayed from Dec 2021).
  It was to be called the 2022 version of EJScreen, and
  here is called bg22.

EJScreen 2.0 was released by EPA 2022-02-18 (delayed from mid/late 2021).
  EJScreen 2.0 used ACS2019, which is from 2015-2019 (released Dec 2020).
 It was to be called the 2021 version, and here is called bg21 as it was to be a late 2021 version.

 This is from the EJSCREEN dataset from the ftp site but with
 fields renamed for easier use in the ejscreen package.
 Versions without renamed columns were USA_2022_LOOKUP and States_2022_LOOKUP.

It can be used with for example ejanalysis::lookup.pctile(13, varname.in.lookup.table = 'pm', look
It shows what the cutpoints are for each variable at percentiles 0,1,2 through 99, 100.
 For example, if the traffic.score is 1000 in a given location,
 you can look where that falls in the percentiles and see that 81
ejanalysis::lookup.pctile(1000, varname.in.lookup.table = 'traffic.score',  lookup = lookupUSA)
```

Details

The EJScreen 2.1 version could be replicated for this package via `x <- ejscreen::ejscreen.lookuptables(bg22)`
The earlier versions were created for this package from the EJSCREEN geodatabase
downloaded from the EJSCREEN public FTP site as .gdb format (zipped).
A script can be used to import and clean it up from that point:
see SCRIPT_read-downloaded-pctile-lookups.R in the inst folder of this pkg

See Also

lookupUSA lookupRegions lookupStates ejscreen.lookuptables ejanalysis::lookup.pctile()

Examples

```
ejanalysis::lookup.pctile(1000, varname.in.lookup.table = 'traffic.score',  lookup = lookupUSA)
    ejanalysis::lookup.pctile(c(1000, 3000), varname.in.lookup.table = 'traffic.score',
      lookup = lookupStates, zone = 'NY')
  ## Not run:
   bg <- bg22[sample(1:NROW(bg22), 100), ]
  state.pctile.pm <- ejanalysis::lookup.pctile(myvector = bg$pm, varname.in.lookup.table = 'pm',
      lookup = lookupStates, zone = bg$ST)
   plot(state.pctile.pm, bg$pctile.pm, pch = '.')
   text(state.pctile.pm, bg$pctile.pm, labels = paste(bg$ST, round(bg$pm,1)), cex = 0.8)
   abline(0,1)
```

```
    lookupStates[lookupStates$PCTILE == 'mean', c('REGION', 'pm')]
    lookupUSA[lookupUSA$PCTILE == 'mean', c('REGION', 'pm')]

## End(Not run)
 ## Not run:
   What is environmental score at given percentile?
 ejanalysis::lookup.pctile(40,'cancer',lookupUSA)
 # [1] 84
 ejanalysis::lookup.pctile(40,'cancer',lookupStates,'WV')
 # [1] 93
 #    What is percentile of given environmental score?
 ejscreen::lookupUSA[lookupUSA$PCTILE=='84' ,'cancer']
 # [1] 39.83055
 ejscreen::lookupStates[lookupStates$PCTILE=='84' & lookupStates$REGION =='WV','cancer']
 # [1] 33.36371
 # also see ejanalysis::assign.pctiles

## End(Not run)
```

`make.popup.d` *Make text to be shown in popups on Demographic data map*

Description

Takes raw values and what percentiles they are at, and presents those as a text field to be used as the text in a popup window on a map

Usage

```
make.popup.d(d, pctile, prefix = "pctile.text.", basenames)
```

Arguments

`d`	raw demographic values, 0-1 (such as 0.3345 where roughly 33 percent of the local population is under age 5)
`pctile`	required integers 0 to 100, representing the percentile(s) at which the raw value(s) fall(s).
`prefix`	optional, default is 'pctile.text.' This is a text string specifying the first part of the desired resulting fieldname in outputs.
`basenames`	optional, default is colnames(d). Defines colname(s) of outputs, which are the prefix plus this.

Details

Note d should be a (vector? or) data.frame of exact demographic percentages from 0 to 1, not 0 to 100 BUT pctile should be INTEGER 0 to 100, NOT 0 to 1! Because that is how EJSCREEN data are stored In EJSCREEN, there are three types of pctile.text fields: E (text varies), D, EJ: 'pctile.text.cancer' "55 lifetime risk per million (91 'pctile.text.pctmin' "13 'pctile.text.EJ.DISPARITY.cancer.eo' "36

Value

Returns character vector or data.frame, same shape as first input parameter.

See Also

make.popup.d make.popup.e make.popup.ej pctileAsText

Examples

```
# inputs are test0 and test1, and desired output is like test2
  # (except note how prefix is added to each basename)
test0 <- structure(list(
  VSI.eo = c(0.185525372063833, 0.174428104575163, 0.485647788983707),
  pctmin = c(0.131656804733727, 0.111928104575163, 0.671062839410395),
  other = c(NA, NA, 0.02)),
  .Names = c("VSI.eo", "pctmin", "other"),
  row.names = c(NA, 3L), class = "data.frame")
 test0
# VSI.eo    pctmin other
# 1 0.1855254 0.1316568    NA
# 2 0.1744281 0.1119281    NA
# 3 0.4856478 0.6710628  0.02

 test1 <- structure(list(
   pctile.VSI.eo = c(27.1991395138354, 24.6836238179206, 72.382419748292),
   pctile.pctmin = c(30.2662374847936, 26.761078397073, 78.2620665123235),
   other = c(NA, NA, 4)),
   .Names = c("pctile.VSI.eo", "pctile.pctmin", "other"),
   row.names = c(NA, 3L), class = "data.frame")
 test1
#   pctile.VSI.eo pctile.pctmin other
# 1      27.19914      30.26624    NA
# 2      24.68362      26.76108    NA
# 3      72.38242      78.26207     4

 test2 <- structure(list(
   pctile.text.VSI.eo = c("19% (27%ile)", "17% (24%ile)","49% (72%ile)"),
   pctile.text.pctmin = c("13% (30%ile)", "11% (26%ile)", "67% (78%ile)"),
   other = c(NA, NA, 4)),
   .Names = c("pctile.text.VSI.eo","pctile.text.pctmin", "other"),
   row.names = c(NA, 3L), class = "data.frame")
 test2
#   pctile.text.VSI.eo pctile.text.pctmin other
# 1       19% (27%ile)       13% (30%ile)    NA
# 2       17% (24%ile)       11% (26%ile)    NA
# 3       49% (72%ile)       67% (78%ile)     4

make.popup.d(test0, test1)
 #   pctile.text.VSI.eo pctile.text.pctmin pctile.text.other
 # 1       19% (27%ile)       13% (30%ile)              <NA>
 # 2       17% (24%ile)       11% (26%ile)              <NA>
 # 3       49% (72%ile)       67% (78%ile)        2% (4%ile)
```

---

make.popup.e *Make text to be shown in popups on Envt data map*

Description

Takes raw values and what percentiles they are at, and presents those as a text field to be used as text in a popup on a map

Usage

```
make.popup.e(e, pctile, prefix = "pctile.text.", basenames, units, sigfigs)
```

Arguments

e	raw environmental indicator values for various locations
pctile	required integers 0 to 100, representing the percentile(s) at which the raw value(s) fall(s).
prefix	optional, default is 'pctile.text.' This is a text string specifying the first part of the desired resulting fieldname in outputs.
basenames	optional, default is colnames(e). Defines colname(s) of outputs, which are the prefix plus this.
units	optional character vector with one per column of e, default is the units used for the latest (2016) version of EJSCREEN environmental indicators, such as 'ppb' and 'ug/m3' – function will try to use units appropriate to basenames, looking in data(popupunits), and use '' (blank) if no match is found.
sigfigs	optional, numeric vector with one per col of e, defining number of significant digits to show in popup, defaulting to rules in EJSCREEN latest (2016) version, or just 2 for basenames not found in data(esigfigs).

Details

Could edit code to NOT put in the units when value is NA?
Could edit code to handle cases like only one row, matrix not df?
Could fix to use only one space when no units

EJSCREEN as of 2015 used 85 pctile.text. fields, for popup text, like "pctile.text.EJ.DISPARITY.pm.eo"
`names(bg2)[grepl('pctile.text', names(bg2) )] length( bg2[1, grepl('pctile.text', names(bg2) )] ) # [1] 85`

In EJSCREEN, there are 3 types of pctile.text fields: E (text varies), D, EJ:
`'pctile.text.cancer' "55 lifetime risk per million (91 'pctile.text.pctmin' "13 'pctile.text.EJ.DISP "36 } For E popups, text includes units:\cr (neuro was only in 2015 version, not later versions of EJSCREEN)\cr\cr \code{ names.e.pctile[names.e.pctile != 'pctile.neuro'] # [1] "pctile.pm" "pctile.o3" "pctile.cancer" # [4] "pctile.resp" "pctile.dpm" "pctile.pctpre1960" # [7] "pctile.traffic.score" "pctile.proximity.npl" "pctile.proximity.rmp" # [10] "pctile.proximity.tsdf "pctile.proximity.npdes" } # NOTE HOW UNITS ARE PART OF THE POPUP, AND IT USES SPECIAL ROUNDING RULES \cr # #' # Stored in data('popunits') # colnames are evar and units \cr\cr \code{ t(bg2[1,gsub('pctile','pctile.text', names.e.pctile[names.e.pctile != 'pctile.neuro'])]) # # pctile.text.pm "10.4 ug/m3 (76 # pctile.text.o3 "42.8 ppb (22%ile)" # pctile.text.cancer "55 lifetime risk per million (91%ile)" # pctile.text.resp "2.1 (72%ile)" # pctile.text.dpm "0.401 ug/m3 (24%ile)" # pctile.text.pctpre1960 "0.4 = fraction pre-1960 (68%ile)" # pctile.text.traf "23 daily vehicles/meters distance (28%ile)" # pctile.text.proximity.npl "0.071 sites/km distance (55%ile)" # pctile.text.proximity.rmp "0.085 facilities/km distance (21%ile)" # pctile.text.proximity.tsdf "0 facilities/km distance (26%ile)" # pctile.text.proximity.npdes "0.25 facilities/km distance (70%ile)" # t(bg2[125:126, gsub('pctile', 'pctile.text', names.e.pctile[names.e.pctile != 'pctile.neuro'])]) # 125 126 # pctile.text.pm "8.37`

ug/m3 (27%ile)" NA # pctile.text.o3 "41.7 ppb (19%ile)" NA # pctile.text.cancer "36 lifetime risk per million (37%ile)" NA # pctile.text.resp "1.4 (37%ile)" NA # pctile.text.dpm "0.275 ug/m3 (13%ile)" NA # pctile.text.pctpre1960 "0.055 = fraction pre-1960 (27%ile)" "0 = fraction pre-1960 (10 # pctile.text.traffic.score "1.7 daily vehicles/meters distance (6 "0 daily vehicles/meters distance (2 # pctile.text.proximity.npl "0.056 sites/km distance (47 "0 sites/km distance (16 # pctile.text.proximity.rmp "0.046 facilities/km distance (7 "0 facilities/km distance (1 # pctile.text.proximity.tsdf "0 facilities/km distance (26 "0 facilities/km distance (26 # pctile.text.proximity.npdes "0.067 facilities/km distance (16 "0 facilities/km distance (1 # # single result, e.g.: "24

Value

Returns character vector or data.frame, same shape as first input parameter.

See Also

esigfigs make.popup.d make.popup.e make.popup.ej pctileAsText

Examples

```
  #   Example: inputs are test0 and test1, and desired output is like test2
  #   (except note how prefix is added to each basename)

test0 <- structure(list(
  e1 = c(0.185525372063833, 0.174428104575163, 0.485647788983707),
  e2 = c(0.131656804733727, 0.111928104575163, 0.671062839410395),
  other = c(NA, NA, 0.02)),
  .Names = c("e1", "e2", "other"),
  row.names = c(NA, 3L), class = "data.frame")
test0

 test1 <- structure(list(
   pctile.e1 = c(27.1991395138354, 24.6836238179206, 72.382419748292),
   pctile.e2 = c(30.2662374847936, 26.761078397073, 78.2620665123235),
   other = c(NA, NA, 4)),
   .Names = c("pctile.e1", "pctile.e2", "other"),
   row.names = c(NA, 3L), class = "data.frame")
 test1

test2 <- structure(list(
  pctile.text.e1 = c("19 (27%ile)", "17 (24%ile)","49 (72%ile)"),
  pctile.text.e2 = c("13 (30%ile)", "11 (26%ile)", "67 (78%ile)"),
  other = c(NA, NA, 4)),
  .Names = c("pctile.text.e1","pctile.text.e2", "other"),
  row.names = c(NA, 3L), class = "data.frame")
test2

## make.popup.e(test0, test1)
```

---

make.popup.ej *Make text to be shown in popups on EJ map*

Description

Takes percentiles (unlike make.popup.d or make.popup.e, which need raw values too), and presents those as a text field to be used as the text in a popup window on a map.

Usage

```
make.popup.ej(pctile, prefix = "pctile.text.", basenames)
```

Arguments

pctile	required integers 0 to 100
prefix	optional, default is 'pctile.text.' This is a text string specifying the first part of the desired resulting fieldname in outputs.
basenames	optional, default is 'pctile.xxx' where xxx is colnames(pctile). Defines colname(s) of outputs, which are the prefix plus this.

Details

Note pctile should be a (vector? or) data.frame of percentiles as INTEGER 0 to 100, NOT 0 to 1! Because that is how EJSCREEN data are stored. Might add code to handle cases like only one row, matrix not df, etc? Assume normal EJSCREEN pctile cols here would be like pctile.EJ.DISPARITY.pm.eo and then output popup col would be like pctile.text.EJ.DISPARITY.pm.eo In EJSCREEN, there are three types of pctile.text fields: E (text varies), D, EJ: 'pctile.text.cancer' "55 lifetime risk per million (91 'pctile.text.pctmin' "13 'pctile.text.EJ.DISPARITY.cancer.eo' "36

Value

Returns character vector or data.frame, same shape as pctile.

See Also

make.popup.d make.popup.e make.popup.ej pctileAsText

Examples

```
test1 <- structure(list(
 pctile.EJ.DISPARITY.pm.eo = c(43.1816682334032, 27.4198086017171, 71.7852110581344, NA),
 pctile.EJ.DISPARITY.o3.eo = c(47.1675935028896, 33.9578650432096, 69.7501760334948, NA)),
  .Names = c("pctile.EJ.DISPARITY.pm.eo", "pctile.EJ.DISPARITY.o3.eo"),
  row.names = c(1L, 2L, 3L, 126L), class = "data.frame")
test1
#    pctile.EJ.DISPARITY.pm.eo pctile.EJ.DISPARITY.o3.eo
#1                    43.18167                  47.16759
#2                    27.41981                  33.95787
#3                    71.78521                  69.75018
#126                        NA                        NA

test2 <- structure(list(
  pctile.text.EJ.DISPARITY.pm.eo = c("43%ile", "27%ile", "71%ile", NA),
  pctile.text.EJ.DISPARITY.o3.eo = c("47%ile", "33%ile", "69%ile", NA)),
  .Names = c("pctile.text.EJ.DISPARITY.pm.eo", "pctile.text.EJ.DISPARITY.o3.eo"),
  row.names = c(1L, 2L, 3L, 126L), class = "data.frame")
test2
#    pctile.text.EJ.DISPARITY.pm.eo pctile.text.EJ.DISPARITY.o3.eo
```

```
#1                          43%ile                          47%ile
#2                          27%ile                          33%ile
#3                          71%ile                          69%ile
#126                          <NA>                            <NA>

make.popup.ej(test1)
#  pctile.text.EJ.DISPARITY.pm.eo pctile.text.EJ.DISPARITY.o3.eo
#1                         43%ile                         47%ile
#2                         27%ile                         33%ile
#3                         71%ile                         69%ile
#4                           <NA>                           <NA>
```

`MeansByGroup_and_Ratios` *average of each environmental indicator in each demographic group and ratio to rest of US*

Description

for the EJScreen 2.1 dataset, average of each environmental indicator in each demographic group and ratio to rest of US

Details

see ejanalysis::RR.means() and ejscreen/inst/SCRIPT_make_MeansByGroup_and_Ratios_RRS.US22.R

`names.d` *Fieldnames of demographic columns in ejscreen package data*

Description

This data set provides variables that hold the colnames of demographic fields in data.frames that may be used in the ejscreen package to make it easier to refer to them as a vector, e.g., mydf[, names.e] Note that earlier versions of EJSCREEN also included fields related to a demographic indicator that used six not two components: VSI.svi6, names.d.svi6, names.d.svi6.bin, names.d.svi6.pctile

Usage

```
data('names.dvars'); names.d
```

Format

A series of variables (each is a character vector of colnames):

- "names.d" (VSI.eo, pctmin, pctlowinc, etc.)
- "names.d.bin"
- "names.d.eo"
- "names.d.eo.bin"
- "names.d.eo.pctile"

- "names.d.pctile"
- "names.d.subgroups"
- "names.d.subgroups.count"
- "names.d.subgroups.pct" #'
- "Dlist" (this one is like names.d, but as a list, not a vector)

Source

Names developed for this package. No external data source.

See Also

ejscreenformulas names.e names.d names.ej

`names.d.nice` *Nicer names for demog fields in ejscreen data*

Description

This data set provides nicer names for the ejscreen demographic indicator variables. These can be used to label graphs, for example.

Format

character vector

Details

Defaults to the latest version

See Also

ejscreenformulas names.e names.d names.ej

`names.e` *Fieldnames of environmental indicator columns in ejscreen package data*

Description

This data set provides variables that hold the colnames of environmental indicator fields in data.frames that may be used in the ejscreen package to make it easier to refer to them as a vector, e.g., mydf[, names.e]

Format

A series of variables (each is a character vector of colnames). For the 2.0 version of EJSCREEN:

- "names.e" (pm, o3, cancer, resp, dpm, pctpre1960, traffic.score, proximity.npl, proximity.rmp, proximity.tsdf, proximity.npdes, ust)
- "names.e.bin"
- "names.e.pctile"
- "Elist" (this one is like names.e, but as a list, not a vector)

Details

NOTE: This used to provide the 2015 version's list, which had "neuro" in it, but now defaults to the latest (2016) version

Source

Names developed for this package. No external data source.

See Also

`names.e.nice` `ejscreenformulas` `names.d` `names.ej`

---

names.e.nice *Nicer names for envt fields in ejscreen data*

Description

This data set provides nicer names for the ejscreen environmental indicator variables. These can be used to label graphs, for example.

Usage

```
data('names.e.nice')
```

Format

character vector

Details

Defaults to the latest version

See Also

`ejscreenformulas` `names.e`

`names.ej` *Fieldnames of environmental justice indicator columns in ejscreen package data*

Description

This data set provides variables that hold the colnames of environmental indicator fields in data.frames that may be used in the ejscreen package to make it easier to refer to them as a vector, e.g., mydf[, names.ej]

Format

A series of variables are available (each is a character vector of colnames):

- "names.ej"
- "names.ej.bin"
- "names.ej.pctile"
- "namesall.ej"
- "namesall.ej.bin"
- "namesall.ej.pctile"

And names.ej in turn is this, for example:

- [1] "EJ.DISPARITY.pm.eo"
- [2] "EJ.DISPARITY.o3.eo"
- [3] "EJ.DISPARITY.cancer.eo"
- [4] "EJ.DISPARITY.resp.eo"
- [5] "EJ.DISPARITY.dpm.eo"
- [6] "EJ.DISPARITY.pctpre1960.eo"
- [7] "EJ.DISPARITY.traffic.score.eo"
- [8] "EJ.DISPARITY.proximity.npl.eo"
- [9] "EJ.DISPARITY.proximity.rmp.eo"
- [10] "EJ.DISPARITY.proximity.tsdf.eo"
- [11] "EJ.DISPARITY.proximity.npdes.eo"
- [12] "EJ.DISPARITY.ust.eo"

Details

This should have the latest version. Old versions also had fields related to svi6 the 6-demographic-variable indicator and other versions of EJ index formula with PCT or BURDEN in the variable name instead of DISPARITY.

Source

Names developed for this package. No external data source.

See Also

`ejscreenformulas` `names.e` `names.d` `names.ej.pctile`

nicenames — *Convert EJSCREEN R variable names to more descriptive labels from glossary*

Description

Provides more descriptive variable names drawn from the ejscreenformulas, using the ejscreenformulas$glossaryfieldname that corresponds to each ejscreenformulas$Rfieldname in x For example it provides " if given "pctlowinc"

Usage

```
nicenames(x)
```

Arguments

x	EJSCREEN variable names as found in ejscreenformulas$Rfieldnames, one or more as character vector

Value

character vector same shape as x, with nice name if avail or unchanged x where otherwise

Examples

```
nicenames(names.e)
nicenames(names.d)
nicenames(names.d.subgroups)
```

pctileAsText — *Utility function in showing a percentile as popup text*

Description

Converts numeric percentiles (0-100) into character (text) that converts 95.3124 to '95

Usage

```
pctileAsText(x)
```

Arguments

x	vector or data.frame of numeric values 0 to 100 (not 0 to 1), representing percentiles from EJSCREEN dataset

Value

Returns matrix/vector of same shape as x if x was data.frame/vector

Examples

```
## Not run:
(bg2[ 125:126, c('pctile.pctmin', 'pctile.EJ.DISPARITY.pm.eo') ])
(bg2[ 125:126, c('pctile.text.pctmin', 'pctile.text.EJ.DISPARITY.pm.eo') ])
pctileAsText(bg2[ 125:126, c('pctile.pctmin', 'pctile.EJ.DISPARITY.pm.eo') ])

## End(Not run)
```

---

pj *plot ej percentiles vs percentiles from EJSCREEN data*

Description

plot ej percentiles vs percentiles from EJSCREEN data

Usage

```
pj(enum, dnum, dat = ejscreen::bg22, ...)
```

Arguments

enum	which of the envt indicators to use, such as 1
dnum	which of the demog indicators to use, such as 3
dat	data.frame of ejscreen data
...	passed to ileile_plot()

See Also

ileile_plot()

Examples

```
pj(1,3)
pj(1,6, col="blue", main="My Graphic")
```

---

popupunits *Units of measurement for environmental indicators*

Description

Table indicating what units to use, such as ug/m3, in showing environmental indicators in EJSCREEN, as shown in popup windows on maps

Usage

```
data('popupunits')
```

Format

A data.frame:

```
> str(popupunits) 'data.frame': 11 obs. of 2 variables: $ evar : chr "pm" "o3" "cancer"
... $ units: chr "ug/m3" "ppb" "lifetime risk per million" "" ... > popupunits evar units
1 pm ug/m3 2 o3 ppb 3 cancer lifetime risk per million 4 resp 5 dpm ug/m3 6 pctpre1960 = fraction
pre-1960 7 traffic.score daily vehicles/meters distance 8 proximity.npl sites/km distance
9 proximity.rmp facilities/km distance 10 proximity.tsdf facilities/km distance 11 proximity.npdes
facilities/km distance
```

Source

See related Technical Documentation at http://www.epa.gov/ejscreen

See Also

make.popup.e names.e

RRS.county *Ratios of mean indicator values across demographic groups by US County*

Description

Based on the latest EJSCREEN dataset

See ejanalysis::RR.table() for how this was created and can be used.

RRS.REGIONS *Ratios of mean indicator values across demographic groups by EPA Region*

Description

Based on the latest EJSCREEN dataset

See ejanalysis::RR.table() for how this was created and can be used.

RRS.ST *Ratios of mean indicator values across demographic groups by State*

Description

Based on the latest EJSCREEN dataset

See ejanalysis::RR.table() for how this was created and can be used.

RRS.US — *Ratios of mean indicator values across demographic groups*

Description

Based on the latest EJSCREEN dataset See ejanalysis::RR.means() and ejscreen/inst/SCRIPT_make_MeansByGroup_an and ejanalysis::RR.table() for how this was created and can be used.

tract22DemographicSubgroups2016to2020 — *Demographic subgroups of race/ethnicity by Census Tract - Missing Puerto Rico*

Description

This dataset fits with EJScreen 2.1, released in late 2022, based on ACS 2016-2020.

```
This data was created by downloading and calculating
DETAILED RACE ETHNICITY SUBGROUP VARIABLES THAT ARE NOT IN EJSCREEN
(the subgroups within "minority" or "people of color").
This dataset includes percent Hispanic,
percent Non-Hispanic Black Alone (not multirace), etc.
Race ethnicity groups are defined by Census Bureau. They are
mutually exclusive (no overlaps between groups,
so a person is always in only one of these groups)
so they add up to the total population count or percent.

From Census ACS 5 year summary file.
```

ustotals — *Get US Totals and Percentages Overall for EJSCREEN Fields*

Description

NOTE: May replace with ustotals2 from batch.summarizer pkg, and or replace to be more generic by using ejscreenformulas style formulas rather than formulas and variable names hard coded in this function.

This function simply takes a data.frame of EJSCREEN demographic data and returns the total count or overall US percentage for various fields, by using the appropriate denominator (universe) to calculate any given percentage. For example, PCTLOWINC.US equals sum(lowinc) / sum(povknownratio), not sum(lowinc) / sum(pop). This function is hard-coded to use specified field names referring to EJSCREEN variables. This function is not needed to create an EJSCREEN dataset, but is convenient if one wants US summary values.

Usage

```
ustotals(bg)
```

Arguments

bg	Must be a data.frame that has the following colnames: • pop, • lowinc, • mins, • under5, • over64, • lths, • lingiso, • pre1960, • hisp, • nhwa, • nhba, • nhaiana, • nhaa, • nhnhpia, • nhotheralone, • nhmulti, • povknownratio, • age25up, • hhlds, • builtunits

Value

Returns a named list of US totals and percentages (as fractions 0-100) (e.g., POP.US=xxxx, etc.):

- POP.US,
- LOWINC.US,
- MINS.US,
- UNDER5.US,
- OVER64.US,
- LTHS.US,
- LINGISO.US,
- POVKNOWNRATIO.US # denominator FOR PCTLOWINC,
- BUILTUNITS.US # denominator FOR PCTPRE1960,
- HHLDS.US # denominator FOR LINGISO,
- PRE1960.US,
- HISP.US,
- NHWA.US,

- NHBA.US,
- NHAIANA.US,
- NHAA.US,
- NHNHPIA.US,
- NHOTHERALONE.US,
- NHMULTI.US,
- PCTLOWINC.US,
- PCTMIN.US,
- PCTUNDER5.US,
- PCTOVER64.US,
- PCTLTHS.US,
- PCTLINGISO.US,
- PCTPRE1960.US,
- PCTHISP.US,
- PCTNHWA.US,
- PCTNHBA.US,
- PCTNHAIANA.US,
- PCTNHAA.US,
- PCTNHNHPIA.US,
- PCTNHOTHERALONE.US,
- PCTNHMULTI.US

Examples

```
## Not run:
# See EJSCREEN demographic variables plus some race/ethnicity subgroups
c1 = ejscreen::ustotals(ejscreen::bg20DemographicSubgroups2014to2018)
c1 = cbind(c1[c1 != 0 & !is.na(c1)])
c2 = ejscreen::ustotals(ejscreen::bg20[!is.na(ejscreen::bg20$ST),])
### c2 = ejscreen::ustotals(ejscreen::bg20[!is.na(ejscreen::bg20$ST) & ejscreen::bg20$ST != 'PR',])
c2 = cbind(c2[c2 != 0 & !is.na(c2)])
US_2014to2018 = unique(rbind(c1, c2))
colnames(US_2014to2018) = 'total_count_or_fraction'
print(US_2014to2018)


 # Display as a nice table with two columns, rounded numbers, rownames and colnames
 tots <- ustotals(bg20)
 tots <- round(cbind(unlist(tots)), 3)
 totrownames <- rownames(tots)[1:16]
 tots <- cbind(tots[1:16], 100 * c(1, tots[17:31]))
 rownames(tots) <- totrownames
 colnames(tots) <- c('count', 'pct')
 tots

## End(Not run)
```

Index