Package 'synthACS'

January 26, 2020

Title Synthetic Microdata and Spatial MicroSimulation Modeling for ACS Data

Version 1.5.3

Description Provides access to curated American Community Survey (ACS) base tables via a wrapper to library(acs). Builds synthetic micro-datasets at any user-specified geographic level with ten default attributes; and, conducts spatial microsimulation modeling (SMSM) via simulated annealing. SMSM is conducted in parallel by default. Lastly, we provide functionality for data-extensibility of micro-datasets.

Depends R (>= 3.2.3)

Imports data.table (>= 1.9.6), Rcpp, acs (>= 2.1)

License MIT + file LICENSE

LazyData true

Suggests testthat, sp (>= 1.2), leaflet (>= 1.0), shiny (>= 0.13), knitr, rmarkdown, R.rsp

RoxygenNote 6.1.1

Encoding UTF-8

LinkingTo Rcpp

VignetteBuilder knitr, R.rsp

NeedsCompilation yes

Author Alex Whitworth [aut, cre]

Maintainer Alex Whitworth <whitworth.alex@gmail.com>

Repository CRAN

Date/Publication 2020-01-26 06:00:02 UTC

R topics documented:

`add_constraint` *Add new constraint to constraint table*

Description

Add a new constraint to the mapping between a given macro dataset (class "macroACS") and a matching micro dataset (class "micro_synthetic). May be called repeatedly to create a set of constraints.

Usage

```
add_constraint(attr_name = "variable", attr_totals, micro_data,
  constraint_list = NULL)
```

Arguments

`attr_name`	The name of the attribute, or variable, that you wish to constrain.
`attr_totals`	A named integer vector of counts per level of the new constraining attribute.
`micro_data`	The micro dataset, of class `"micro_synthetic"`, for which you wish to add a constraint.
`constraint_list`	A `list` of prior constraints on the same dataset which you wish to add to. Defaults to `NULL` (ie. the default is that this is the first constraint.)

Value

A list of constraints.

Examples

```
## Not run:
## assumes that you have a micro_synthetic dataset named test_micro and attribute counts
## named a,e,g respectively
c_list <- add_constraint(attr_name= "age", attr_totals= a, micro_data= test_micro)
c_list <- add_constraint(attr_name= "edu_attain", attr_totals= e, micro_data= test_micro,
                         constraint_list= c_list)
c_list <- add_constraint(attr_name= "gender", attr_totals= g, micro_data= test_micro,
                          constraint_list= c_list)

## End(Not run)
```

adjDR *Age-adjusted Death Rate by race and gender*

Description

A dataset containing age-adjusted death rate data by race and gender of the deceased. Data is provided for 1980-2013.

Usage

```
adjDR
```

Format

A `data.frame` with 612 observations and 4 variables.

year The year for which data was was recorded.

race The racial group of the deceased One of `all` all races; `white` whites; `black_aa` black / African-American; `nat_amer` American Indian or Native Alaskan; `asian_isl` Asian or Pacific Islander; `hisp_lat` Hispanic.

gender The gender of the deceased. One of `c(both,male,female)`

adj_death_rate The age-adjusted death rate. See details.

Details

- The age-adjusted death rates are used to compare relative mortality risks among groups and over time. They were computed by the direct method, which is defined

$$R' = \sum_i \frac{P_{si}}{P_s} R_i$$

  where P_{si} is the standard population for age group i, P_s is the total US standard population and R_i is the raw death rate for age group i.

- Populations are based on census counts enumerated as of April 1 of the census year and estimated as of July 1 for non-census years.

Source

http://www.cdc.gov/nchs/nvss/deaths.htm

References

Xu, J. Q., S. L. Murphy, and K. D. Kochanek. "Deaths: final data for 2013." National Vital Statistics Reports 64.2 (2015).

AgeRaceDR *Death rates in the United States by age and race, 2013*

Description

A dataset containing death rates for individuals by age group and race for the United States, 2013.

Usage

```
AgeRaceDR
```

Format

A `data.frame` with 360 observations and 4 variables.

age The exact age, in years, at which life expectany is calculated.

race The racial group of the deceased One of `all` all races; `white` whites; `black` black / African-American; `hispanic` Hispanic; `asian.isl` Asian and Pacific Islander; `nat.amer` Native American or Alaska Native.

gender The gender of the deceased. One of `c(both,male,female)`

death_rate The raw death rate. See details.

Details

- The death rate is defined as deaths per 100,000 population.

Source

http://www.cdc.gov/nchs/nvss/deaths.htm

References

Xu, J. Q., S. L. Murphy, and K. D. Kochanek. "Deaths: final data for 2013." National Vital Statistics Reports 64.2 (2015).

all_geogs_add_constraint *Add new constraint to a set of geographies*

Description

Add a new constraint to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'macro_micro'). May be called repeatedly to create a set of constraints across the sub-geographies.

Usage

```
all_geogs_add_constraint(attr_name = "variable", attr_total_list,
  macro_micro, constraint_list_list = NULL)
```

Arguments

attr_name	The name of the attribute, or variable, that you wish to constrain.
attr_total_list	A list of named integer vectors containing counts per level of the new constraining attribute for each geography.
macro_micro	The geographical dataset of macro and micro data. Should be of class "macro_micro".
constraint_list_list	A list of lists containing prior constraints on the same dataset for which you wish to add to. Defaults to NULL (ie. the default is that this is the first constraint.)

Value

A list of constraint lists.

See Also

add_constraint

Examples

```
## Not run:
# assumes that micro_synthetic already exists in your environment

# 1. build constraints for gender and age
g <- all_geog_constraint_gender(micro_synthetic, method= "macro.table")

a <- all_geog_constraint_age(micro_synthetic, method= "macro.table")

# 2. bind constraints to geographies and macro-data
cll <- all_geogs_add_constraint(attr_name= "age", attr_total_list= a,
          macro_micro= micro_synthetic)
cll <- all_geogs_add_constraint(attr_name= "gender", attr_total_list= g,
          macro_micro= micro_synthetic, constraint_list_list= cll)


## End(Not run)
```

all_geog_constraint_age

Create age constraint list to a set of geographies

Description

Create a new age constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_age(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 a1 <- all_geog_constraint_age(obj, "synthetic")
 a2 <- all_geog_constraint_age(obj, "macro_table")

## End(Not run)
```

all_geog_constraint_edu

Create educational attainment constraint list to a set of geographies

Description

Create a new educational attainment constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_edu(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 e1 <- all_geog_constraint_edu(obj, "synthetic")
 e2 <- all_geog_constraint_edu(obj, "macro_table")

## End(Not run)
```

---

all_geog_constraint_employment

Create employment status constraint list to a set of geographies

---

Description

Create a new employment status constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_employment(obj, method = c("synthetic",
  "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 e1 <- all_geog_constraint_employment(obj, "synthetic")
 e2 <- all_geog_constraint_employment(obj, "macro_table")

## End(Not run)
```

`all_geog_constraint_gender` *Create gender constraint list to a set of geographies*

Description

Create a new gender constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_gender(obj, method = c("synthetic", "macro.table"))
```

Arguments

`obj`	An object of class `"synthACS"`.
`method`	One of `c("synthetic","macro.table")`. Specifying `"synthetic"` indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying `"macro.table"` indicates that the constraints are build from the data in the base ACS tables.

See Also

`all_geogs_add_constraint`

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 g1 <- all_geog_constraint_gender(obj, "synthetic")
 g2 <- all_geog_constraint_gender(obj, "macro_table")

## End(Not run)
```

all_geog_constraint_geog_mob

Create geographic mobility constraint list to a set of geographies

Description

Create a new geographic mobility constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_geog_mob(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 gm1 <- all_geog_constraint_geog_mob(obj, "synthetic")
 gm2 <- all_geog_constraint_geog_mob(obj, "macro_table")

## End(Not run)
```

all_geog_constraint_income

Create individual income constraint list to a set of geographies

Description

Create a new individual income constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_income(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 i1 <- all_geog_constraint_income(obj, "synthetic")
 i2 <- all_geog_constraint_income(obj, "macro_table")

## End(Not run)
```

---

all_geog_constraint_marital_status

Create marital status constraint list to a set of geographies

---

Description

Create a new marital status constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_marital_status(obj, method = c("synthetic",
  "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 m1 <- all_geog_constraint_marital_status(obj, "synthetic")
 m2 <- all_geog_constraint_marital_status(obj, "macro_table")

## End(Not run)
```

`all_geog_constraint_nativity` *Create nativity status constraint list to a set of geographies*

Description

Create a new nativity status constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_nativity(obj, method = c("synthetic", "macro.table"))
```

Arguments

`obj`	An object of class `"synthACS"`.
`method`	One of `c("synthetic","macro.table")`. Specifying `"synthetic"` indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying `"macro.table"` indicates that the constraints are build from the data in the base ACS tables.

See Also

`all_geogs_add_constraint`

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 n1 <- all_geog_constraint_nativity(obj, "synthetic")
 n2 <- all_geog_constraint_nativity(obj, "macro_table")

## End(Not run)
```

all_geog_constraint_poverty

Create poverty status constraint list to a set of geographies

Description

Create a new poverty status constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_poverty(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 p1 <- all_geog_constraint_poverty(obj, "synthetic")
 p2 <- all_geog_constraint_poverty(obj, "macro_table")

## End(Not run)
```

all_geog_constraint_race

Create race constraint list to a set of geographies

Description

Create a new race constraint list to the mapping between a a set of macro datasets and a matching set of micro dataset (supplied as class 'synthACS').

Usage

```
all_geog_constraint_race(obj, method = c("synthetic", "macro.table"))
```

Arguments

obj	An object of class "synthACS".
method	One of c("synthetic","macro.table"). Specifying "synthetic" indicates that constraints are built by marginalizing the synthetic micro datasets. Specifying "macro.table" indicates that the constraints are build from the data in the base ACS tables.

See Also

all_geogs_add_constraint

Examples

```
## Not run:
 # assumes that obj of class 'synthACS' already exists in your environment
 r1 <- all_geog_constraint_race(obj, "synthetic")
 r2 <- all_geog_constraint_race(obj, "macro_table")

## End(Not run)
```

---

all_geog_optimize_microdata

Optimize the selection of a micro data population for a set of geographies.

Description

Optimize the candidate micro datasets such that the lowest loss against the macro dataset constraints are obtained. Loss is defined here as total absolute error (TAE) and constraints are defined by the constraint_list_list. Optimization is done by simulated annealing and geographies are run in parallel.

Usage

```
all_geog_optimize_microdata(macro_micro, prob_name = "p",
  constraint_list_list, p_accept = 0.4, max_iter = 10000L,
  seed = sample.int(10000L, size = 1, replace = FALSE), verbose = TRUE)
```

Arguments

macro_micro	The geographical dataset of macro and micro data. Should be of class "macro_micro".
prob_name	It is assumed that observations are weighted and do not have an equal probability of occurance. This string specifies the variable within each dataset that contains the probability of selection.
constraint_list_list	A list of constraint lists. See add_constraint, all_geogs_add_constraint

p_accept	The acceptance probability for the Metropolis acceptance criteria.
max_iter	The maximum number of allowable iterations. Defaults to 10000L
seed	A seed for reproducibility. See set.seed
verbose	Logical. Do you wish to see verbose output? Defaults to TRUE

See Also

optimize_microdata

Examples

```
## Not run:
 # assumes that micro_synthetic and cll already exist in your environment
 # see: examples for derive_synth_datasets() and all_geogs_add_constraint()
 optimized_la <- all_geog_optimize_microdata(micro_synthetic, prob_name= "p",
     constraint_list_list= cll, p_accept= 0.01, max_iter= 1000L)

## End(Not run)
```

---

all_geog_synthetic_new_attribute

Add a new attribute to a set (ie list) of synthetic_micro datasets

---

Description

Add a new attribute to a set (ie list) of synthetic_micro datasets using conditional relationships between the new attribute and existing attributes (eg. wage rate conditioned on age and education level). The same attribute is added to *each* synthetic_micro dataset, where each dataset is supplied a distinct relationship for attribute creation.

Usage

```
all_geog_synthetic_new_attribute(df_list, prob_name = "p",
  attr_name = "variable", conditional_vars = NULL, st_list = NULL)
```

Arguments

df_list	A list of R objects each of class "synthetic_micro".
prob_name	A string specifying the column name of each data.frame in df_list containing the probabilities for each synthetic observation.
attr_name	A string specifying the desired name of the new attribute to be added to the data.
conditional_vars	An character vector specifying the existing variables, if any, on which the new attribute (variable) is to be conditioned on for each dataset. Variables must be specified in order. Defaults to NULL ie- an unconditional new attribute.

st_list — A list of equal length to df_list. Each element of st_list is a data.frame symbol table with N + 2 columns. The last two columns must be: 1. A vector containing the new attribute counts or percentages; 2. is a vector of the new attribute levels. The first N columns must match the conditioning scheme imposed by the variables in conditional_vars. See synthetic_new_attribute and examples.

Value

A list of new synthetic_micro datasets each with class "synthetic_micro".

See Also

synthetic_new_attribute

Examples

```
## Not run:
 set.seed(567L)
 df <- data.frame(gender= factor(sample(c("male", "female"), size= 100, replace= TRUE)),
                 age= factor(sample(1:5, size= 100, replace= TRUE)),
                 pov= factor(sample(c("lt_pov", "gt_eq_pov"),
                                    size= 100, replace= TRUE, prob= c(.15,.85))),
                 p= runif(100))
df$p <- df$p / sum(df$p)
class(df) <- c("data.frame", "micro_synthetic")

# and example test elements
cond_v <- c("gender", "pov")
levels <- c("employed", "unemp", "not_in_LF")
sym_tbl <- data.frame(gender= rep(rep(c("male", "female"), each= 3), 2),
                      pov= rep(c("lt_pov", "gt_eq_pov"), each= 6),
                      cnts= c(52, 8, 268, 72, 12, 228, 1338, 93, 297, 921, 105, 554),
                      lvls= rep(levels, 4))



df_list <- replicate(10, df, simplify= FALSE)
st_list <- replicate(10, sym_tbl, simplify= FALSE)

# run
library(parallel)
syn <- all_geog_synthetic_new_attribute(df_list, prob_name= "p", attr_name= "variable",
                                        conditional_vars= cond_v,st_list= st_list)

## End(Not run)
```

BR2014 *Birth Rates by Age and Race of Mother*

Description

A dataset containing birth rate data in the United States by age and race of the mother. Data for all races is provided for 1970-2014 and for individual races from 1989-2014.

Usage

```
BR2014
```

Format

A `data.frame` with 1,750 observations and 4 variables.

year The year for which data was was recorded.

race The racial group of the mothers. One of `all` all races; `white` non-hispanic whites; `black_aa` black / African-American; `nat_amer` American Indian or Native Alaskan; `asian_isl` Asian or Pacific Islander; `hisp_lat` Hispanic or Latin American.

age_group The age group of the mother.

birth_rate The birth rate. See Details.

Details

- The birth rate is defined as births per 1,000 women in the specified group (age and race).
- Populations are based on census counts enumerated as of April 1 of the census year and estimated as of July 1 for non-census years.
- Beginning in 1997, birth rates for age group 45up by relating births to all women age 45 or older to this group. Prior to 1997, only births to women age 45-49 were included.

Source

http://www.cdc.gov/nchs/nvss/births.htm

References

Hamilton, Brady E., et al. "Births: final data for 2014." National Vital Statistics Reports 64.12 (2015): 1-64.

`calculate_TAE` *Calculate the total absolute error (TAE) between sample data and constraints.*

Description

Calculates the total absolute error (TAE) between sample micro data and constraining totals from the matching macro data. Allows for updating of prior TAE instead of re-calculating to improve speed in iterating. The updating feature is particularly helpful for optimizing micro data fitting via simulated annealing (see `optimize_microdata`).

Usage

```
calculate_TAE(sample_data, constraint_list, prior_sample_totals = NULL,
  dropped_obs_totals = NULL, new_obs = NULL)
```

Arguments

`sample_data`	A `data.frame` with attributes matching `constraint_list`.
`constraint_list`	A `list` of constraints. See `add_constraint`.
`prior_sample_totals`	An optional `list` containing attribute counts of a prior sample corresponding to the constraint list. Defaults to `NULL`.
`dropped_obs_totals`	An optional `list` containing attribute counts from the dropped observations in a prior sample. Defaults to `NULL`.
`new_obs`	An optional `data.frame` containing new observations with attributes matching those in `sample_data`, `constraint_list`, and `prior_sample_totals`. Defaults to `NULL`.

Examples

```
## Not run:
## assumes that you have a micro_synthetic dataset named test_micro and attribute count
## named g respectively
c_list <- add_constraint(attr_name= "gender", attr_totals= g, micro_data= test_micro,
            constraint_list= c_list)
calculate_TAE(test_micro, c_list)

## End(Not run)
```

combine_smsm *Combine separate SMSM optimizations*

Description

Combine objects of class "smsm_set" into a single object of class "smsm_set"

Usage

```
combine_smsm(...)
```

Arguments

...	A list of objects of class 'smsm_set'.

See Also

split, all_geog_optimize_microdata

Examples

```
## Not run:
 combined <- combine_smsm(smsm1, smsm2, smsm3)

## End(Not run)
```

derive_synth_datasets *Derive synthetic micro datasets for a given geography.*

Description

Derive synthetic micro datasets for each sub-geography of a given set of geographic macro data constraining tabulations. See Details... By default, micro dataset generation is run in parallel with load balancing. Macro data is assumed to have been pulled from the US Census API via the acs package.

Usage

```
derive_synth_datasets(macro_data, parallel = TRUE, leave_cores = 2)
```

Arguments

macro_data	A macro dataset list: the result of pull_synth_data.
parallel	Logical, defaults to TRUE. Do you wish to run the operation in parallel?
leave_cores	How many cores do you wish to leave open to other processing?

Value

A `list` of the input macro datasets produced by `pull_synth_data` and a `list` of synthetic micro datasets for each geographical subset within the specified macro geography.

Details

In the absence of true micro level datasets for a given geographic area, synthetic datasets can be used. This function uses conditional and marginal probability distributions (at the aggregate level) to generate synthetic micro population datasets, which are built one constraint at a time. Taking as input the macro level data (class `"macroACS"`), this function builds synthetic micro datasets for each lower level geographical area within the area of study.

In simplest terms, the goal is to generate a joint probability distribution for an attribute vector; and, to create synthetic individuals from this distribution. However, note that information for the full joint distribution is typically not available, so we construct it as a product of conditional and marginal probabilities. This is done one attribute at a time; where it is assumed that there is some sort of continuum of attribute dependence. That is, some attributes are more important (eg. gender, age) in 'determining' others (eg. educational attainment, marital status, etc). These more important attributes need to be assigned first, whereas less important attributes may be assigned later. Most of these distinctions are largely intuitive, but care must be taken in choosing the order of constructed attributes.

This function provides a synthetic population with the following characteristics as well as each synthetic individual's probability of inclusion. The included characteristics are: age, gender, marital status, educational attainment, employment status, nativity, poverty status, geographic mobility in the prior year, individual income, and race. **Note** that these are INDIVIDUAL attributes; they are not at the HOUSEHOLD level. Additional attributes which interest the user may be added in a similar manner via `synthetic_new_attribute`.

References

Birkin, Mark, and M. Clarke. "SYNTHESIS-a synthetic spatial information system for urban and regional analysis: methods and examples." Environment and planning A 20.12 (1988): 1645-1671.

See Also

`pull_synth_data`, `acs.fetch`, `geo.make`

Examples

```
## Not run:
# make geography
la_geo <- acs::geo.make(state= "CA", county= "Los Angeles", tract= "*")
# pull data elements for creating synthetic data
la_dat <- pull_synth_data(2014, 5, la_geo)
# derive synthetic data
la_synthetic <- derive_synth_datasets(la_dat, leave_cores= 0)

## End(Not run)
```

fetch_data *Get Aggregate Data Specified Geography*

Description

Gets aggregate, macro, data, either estimate or standard error, for a specified geography and specified dataset.

Usage

```
fetch_data(acs, geography, dataset = c("estimate", "st.err"),
  choice = NULL)
```

Arguments

acs	An object of class "macroACS".
geography	A character vector allowing string matching via grep to a set of specified geographies. All values may be specified by "*".
dataset	Either "estimate" or "st.err". Do you want data on estimated population counts or estimated standard errors?
choice	A character vector specifying the name of one of the datasets in acs

gen_attr_vectors *Generate attribute vectors*

Description

Generate a list of attribute vectors for new synthetic attribute creation from a "macroACS" object.

Usage

```
gen_attr_vectors(acs, choice)
```

Arguments

acs	An object of class "macroACS".
choice	A character vector specifying the name of one of the datasets in acs

See Also

all_geog_synthetic_new_attribute, synthetic_new_attribute

get_best_fit *Extract best fit for a specified geogrpahy from an 'smsm_set' object*

Description

Extract the best fit micro population (resulting from the simulated annealing algorithm) for a given geography.

Usage

```
get_best_fit(obj, geography)
```

Arguments

obj	An object of class 'smsm_set', typically a result of call to all_geog_optimize_microdata
geography	A string allowing string matching via grep to a specified geography.

get_dataset_names *Get dataset names from a "macroACS" object.*

Description

Get the names of the datasets in a given "macroACS" object.

Usage

```
get_dataset_names(acs)
```

Arguments

acs	An object of class "macroACS".

See Also

fetch_data

get_endyear — *Get the endyear from a "macroACS" object.*

Description

Get the data collection endyear from a "macroACS" object

Usage

```
get_endyear(acs)
```

Arguments

acs	An object of class "macroACS".

get_final_tae — *Extract the final TAE for a specified geogrpahy from an 'smsm_set' object*

Description

Extract the final TAE (resulting from the simulated annealing algorithm) for a given geography.

Usage

```
get_final_tae(obj, geography)
```

Arguments

obj	An object of class 'smsm_set', typically a result of call to all_geog_optimize_microdata
geography	A string allowing string matching via grep to a specified geography.

get_geography — *Get the geography title from a "macroACS" object.*

Description

Get the summary information of the geography selected from a "macroACS" object

Usage

```
get_geography(acs)
```

Arguments

acs	An object of class "macroACS".

get_span *Get the span from a "macroACS" object.*

Description

Get the data collection span from a "macroACS" object

Usage

```
get_span(acs)
```

Arguments

acs	An object of class "macroACS".

is.macroACS *Check macroACS class*

Description

Function that checks if the target object is a macroACS object.

Usage

```
is.macroACS(x)
```

Arguments

x	any R object.

Value

Returns TRUE if its argument has class "macroACS" among its classes and FALSE otherwise.

is.macro_micro *Check macro_micro class*

Description

Function that checks if the target object is a `macro_micro` object.

Usage

```
is.macro_micro(x)
```

Arguments

`x`	any R object.

Value

Returns `TRUE` if its argument has class "macro_micro" among its classes and `FALSE` otherwise.

is.micro_synthetic *Check micro_synthetic class*

Description

Function that checks if the target object is a `micro_synthetic` object.

Usage

```
is.micro_synthetic(x)
```

Arguments

`x`	any R object.

Value

Returns `TRUE` if its argument has class "micro_synthetic" among its classes and `FALSE` otherwise.

is.smsm_set *Check smsm_set class*

Description

Function that checks if the target object is a `smsm_set` object.

Usage

```
is.smsm_set(x)
```

Arguments

`x`	any R object.

Value

Returns `TRUE` if its argument has class "macroACS" among its classes and `FALSE` otherwise.

is.synthACS *Check synthACS class*

Description

Function that checks if the target object is a `synthACS` object.

Usage

```
is.synthACS(x)
```

Arguments

`x`	any R object.

Value

Returns `TRUE` if its argument has class "synthACS" among its classes and `FALSE` otherwise.

la_hospitals *Hospitals in Los Angeles County, CA USA*

Description

An anonymized dataset containing the geographic information of hospitals in Los Angeles County California, USA.

Usage

```
la_hospitals
```

Format

A `data.frame` with 631 observations and 7 variables

geo_long The hospital's longitude.

geo_lat The hospital's lattitude.

city The hospital's postal city.

state_fips The hospital's alpha FIPS code.

zip The hospital's five digit postal ZIP code.

census_tract The census tract in which the hospital is located.

county_name The hospital's county – "LOS ANGELES".

LifeExp *Life expectancy at certain ages; United States, 2013*

Description

A dataset containing life expectancy at certain ages by race, hispanic origin and sex for the United States, 2013.

Usage

```
LifeExp
```

Format

A `data.frame` with 396 observations and 4 variables.

age The exact age, in years, at which life expectany is calculated.

race The racial group of the deceased One of `all` all races; `white` whites; `black` black / African-American; `hispanic` Hispanic; `non.hisp.white` non Hispanic whites; `non.hispanic.black` non Hispanic blacks.

gender The gender of the deceased. One of `c(both,male,female)`

life_expectancy The life expectancy for an individual at the exact age with the given race and gender.

Source

http://www.cdc.gov/nchs/nvss/deaths.htm

References

Xu, J. Q., S. L. Murphy, and K. D. Kochanek. "Deaths: final data for 2013." National Vital Statistics Reports 64.2 (2015).

`marginalize_attr` *Marginalize synthetic attributes*

Description

Marginalize, (ie- reduce in number), attributes of a synthetic dataset of class 'micro_synthetic' or a list of synthetic datasets of class 'synthACS'. This is done by marginalizing the joint distribution based on a set of specified attributes (see Arguments below).

Usage

```
marginalize_attr(obj, varlist, marginalize_out = FALSE)
```

Arguments

`obj`	An object of class `"micro_synthetic"`.
`varlist`	A character vector of variable, or attribute, names in `obj`.
`marginalize_out`	Logical. Do you wish to *remove* the variables in `varlist` instead of keeping them? Defaults to `FALSE`

Examples

```
{
# dummy data setup
set.seed(567L)
df <- data.frame(gender= factor(sample(c("male", "female"), size= 100, replace= TRUE)),
                 age= factor(sample(1:5, size= 100, replace= TRUE)),
                 pov= factor(sample(c("below poverty", "at above poverty"),
                                    size= 100, replace= TRUE, prob= c(.15,.85))),
                 p= runif(100))
df$p <- df$p / sum(df$p)
class(df) <- c("data.frame", "micro_synthetic")

df2 <- marginalize_attr(df, varlist= "gender")
df3 <- marginalize_attr(df, varlist= c("gender", "age"))
df4 <- marginalize_attr(df, varlist= c("gender", "age"), marginalize_out= TRUE)

df_list <- replicate(10, df, simplify= FALSE)
dummy_list <- replicate(10, list(NULL), simplify= FALSE)
```

```
df_list <- mapply(function(a,b) {return(list(a, b))}, a= dummy_list, b= df_list, SIMPLIFY = FALSE)
class(df_list) <- c("list", "synthACS")

# run the function
df_list2 <- marginalize_attr(df_list, varlist= c("gender", "age"))
}
```

MBR *Multiple Birth Rate data by year and race of mother*

Description

A dataset containing multiple birth rate data by race of the mother. Data for all races is provided for 1980-2014 and for individual races from 1990-2014.

Usage

```
MBR
```

Format

A `data.frame` with 110 observations and 8 variables.

year The year for which data was was recorded.

race The racial group of the mothers. One of `all` all races; `white` non-hispanic whites; `black_aa` non Hispanic black / African-American; `hisp_lat` Hispanic.

births Total births for the year and racial group in the United States.

twin_births Total twin births for the year and racial group in the United States.

triplet_more_births Total triplet or higher order births for the year and racial group in the United States.

MBRate The number of live births in all multiple deliveries per 1,000 live births.

twinBR The number of live births in all twin deliveries per 1,000 live births.

twinBR The number of live births in all triplet or higher order deliveries per 100,000 live births.

Details

- Data for race cateogry `"all"` includes races other than white and black and origin not stated.
- Race and Hispanic origin are reported separately on birth certificates. Persons of Hispanic origin may be of any race.

Source

http://www.cdc.gov/nchs/nvss/births.htm

References

Hamilton, Brady E., et al. "Births: final data for 2014." National Vital Statistics Reports 64.12 (2015): 1-64.

`optimize_microdata` *Optimize the selection of a micro data population.*

Description

Optimize the candidate micro dataset such that the lowest loss against the macro dataset constraints is obtained. Loss is defined here as total absolute error (TAE) and constraints are defined by the `constraint_list`. Optimization is done by simulated annealing–see details.

Usage

```
optimize_microdata(micro_data, prob_name = "p", constraint_list,
  tolerance = round(sum(constraint_list[[1]])/2000 *
  length(constraint_list), 0),
  resample_size = min(sum(constraint_list[[1]]), max(500,
  round(sum(constraint_list[[1]]) * 0.005, 0))), p_accept = 0.4,
  max_iter = 10000L, seed = sample.int(10000L, size = 1, replace =
  FALSE), verbose = TRUE)
```

Arguments

`micro_data`	A `data.frame` of micro data observations.
`prob_name`	It is assumed that observations are weighted and do not have an equal probability of occurance. This string specifies the variable within `micro_data` that contains the probability of selection.
`constraint_list`	A `list` of constraining macro data attributes. See `add_constraint`
`tolerance`	An integer giving the maximum acceptable loss (TAE), enabling early stopping. Defaults to a misclassification rate of 1 individual per 1,000 per constraint.
`resample_size`	An integer controlling the rate of movement about the candidate space. Specifically, it specifies the number of observations to change between iterations. Defaults to `0.5%` the number of observations.
`p_accept`	The acceptance probability for the Metropolis acceptance criteria.
`max_iter`	The maximum number of allowable iterations. Defaults to `10000L`
`seed`	A seed for reproducibility. See `set.seed`
`verbose`	Logical. Do you wish to see verbose output? Defaults to `TRUE`

Details

Spatial microsimulation involves the study of individual-level phenomena within a specified set of geographies in which these individuals act. It involves the creation of synthetic data to model, via simulation, these phenomena. As a first step to simulation, an appropriate micro-level (ie. individual) dataset must be generated. This function creates such appropriate micro-level datasets given a set of candidate observations and macro-level constraints.

Optimization is done via simulated annealing, where we wish to minimize the total absolute error (TAE) between the micro-data and the macro-constraints. The annealing procedure is controlled by the parameters `tolerance`, `resample_size`, `p_accept`, and `max_iter`. Specifically, `tolerance` indicates the maximum allowable TAE between the output micro-data and the macro-constraints within a given `max_iter` allowable iterations to converge. `resample_size` and `p_accept` control movement about the candidate space. Specfically, `resample_size` controls the jump size between neighboring candidates and `p_accept` controls the hill-climbing rate for exiting local minima.

Please see the references for a more detailed discussion of the simulated annealing procedure.

References

Ingber, Lester. "Very fast simulated re-annealing." Mathematical and computer modelling 12.8 (1989): 967-973.

Metropolis, Nicholas, et al. "Equation of state calculations by fast computing machines." The journal of chemical physics 21.6 (1953): 1087-1092.

Szu, Harold, and Ralph Hartley. "Fast simulated annealing." Physics letters A 122.3 (1987): 157-162.

Examples

```
## Not run:
## assumes you have micro_synthetic object named test_micro and constraint_list named c_list
opt_data <- optimize_microdata(test_micro, "p", c_list, max_iter= 10, resample_size= 500,
              p_accept= 0.01, verbose= FALSE)

## End(Not run)
```

---

plot_TAEpath *Plot simulated annealing path*

Description

Plot the path TAE in the simulated annealing algorithm for a given geography

Usage

```
plot_TAEpath(object, geography, ...)
```

Arguments

`object`	An object of class `'smsm_set'`, typically a result of call to `all_geog_optimize_microdata`
`geography`	A string allowing string matching via `grep` to a specified geography.
`...`	additional arguments passed to other methods

`pull_acs_basetables` *Pull ACS base tables*

Description

A wrapper function to pull multiple base tables from ACS API via `acs.fetch`.

Usage

```
pull_acs_basetables(endyear, span, geography, table_vec)
```

Arguments

`endyear`	An integer, indicating the latest year of the data in the survey.
`span`	An integer in `c(1,3,5)` indicating the span of the desired data.
`geography`	a valid `geo.set` object specifying the census geography or geographies to be fetched.
`table_vec`	A `character` vector specifying ACS base tables.

Value

A `'macroACS'` class object

References

https://www.census.gov/programs-surveys/acs/technical-documentation/summary-file-documentation.html

Examples

```
## Not run:
# make geography
la_geo <- acs::geo.make(state= "CA", county= "Los Angeles City")
# pull data
la_dat <- pull_acs_basetables(endyear= 2015, span= 1, geography= la_geo,
  table_vec= c("B00001", "B00002", "B01003"))

## End(Not run)
```

pull_bachelors *Pull ACS data on field of bachelor's degree*

Description

Pull ACS data for a specified geography from base tables B15011 and B15012. Note: only 2014 data is supplied by ACS

Usage

```
pull_bachelors(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

pull_edu *Pull ACS educational attainment and enrollment data*

Description

Pull ACS data for a specified geography from base tables B14001, B14003, B15001, B15002. Not currently implemented: B15010, B28006 Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_edu(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A `list` containing the endyear, span, a `data.frame` of estimates, a `data.frame` of standard errors, a character vector of the original column names, and a `data.frame` of the geography metadata from `acs.fetch`.

See Also

`acs.fetch`, `geo.make`

`pull_geo_mobility` *Pull ACS geographic mobility data*

Description

Pull ACS data for a specified geography from base tables B07001, B07003, B07008, B07009, B07010, and B07012. These tables provide data on geographic mobility in the past year by a number of slices. Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_geo_mobility(endyear, span, geography)
```

Arguments

`endyear`	An integer, indicating the latest year of the data in the survey.
`span`	An integer in `c(1,3,5)` indicating the span of the desired data.
`geography`	a valid `geo.set` object specifying the census geography or geographies to be fetched.

Value

A `list` containing the endyear, span, a `data.frame` of estimates, a `data.frame` of standard errors, a character vector of the original column names, and a `data.frame` of the geography metadata from `acs.fetch`.

See Also

`acs.fetch`, `geo.make`

pull_household *Pull ACS data on households and housing units*

Description

Pull ACS data for a specified geography from base tables B09019, B11011, B19081, B25002, B25003, B25004, B25010, B25024, B25056, B25058, B25071, and B27001. Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_household(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make B28001 - TYPES OF COMPUTERS IN HOUSEHOLD B28002 - PRESENCE AND TYPES OF INTERNET SUBSCRIPTIONS IN HOUSEHOLD

pull_inc_earnings *Pull ACS income and earnings data*

Description

Pull ACS data for a specified geography from base tables B19083, B19301, B19326, B21001, B22001, B23020, B24011. Not yet implemented: B28004 Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_inc_earnings(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

---

pull_mar_status *Pull ACS marital status data*

Description

Pull ACS data for a specified geography from base tables B12001, B12006, B12007, 12501 Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_mar_status(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

pull_population *Pull ACS population data*

Description

Pull ACS data for a specified geography from base tables B01001, B01002, B02001, B06007, B06008, B06009, B06010, B06011, AND B06012. These tables reference population counts by a number of slices. Multiple additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_population(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

pull_pov_inc *Pull ACS income and earnings data*

Description

Pull ACS data for a specified geography from base tables B17001, B17004, B18101, B19001, B19013, B19055, B19057. Not yet implemented: B17002 Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_pov_inc(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

---

pull_race_data *Pull ACS race data*

Description

Pull ACS data for a specified geography from base tables B01001B-I and B02001. ' These tables reference population counts by race.

Usage

```
pull_race_data(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

pull_synth_data *Pull ACS data for synthetic data creation.*

Description

Pull ACS data for a specified geography from base tables B01001, B02001, B12002, B15001, B06001, B06010, B23001, B17005, and B17005. These tables reference population counts by a number of slices. Multiple additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_synth_data(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a list of data.frames of estimates, a list of data.frames of standard errors, and the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

Examples

```
## Not run:
# make geography
la_geo <- acs::geo.make(state= "CA", county= "Los Angeles", tract= "*")
# pull data elements for creating synthetic data
la_dat <- pull_synth_data(2014, 5, la_geo)

## End(Not run)
```

pull_transit_work *Pull ACS transit and work data*

Description

Pull ACS data for a specified geography from base tables B08012, B08101, B08121, B08103, B08124, B08016, B08017. Additional fields, mainly percentages and aggregations, are calculated.

Usage

```
pull_transit_work(endyear, span, geography)
```

Arguments

endyear	An integer, indicating the latest year of the data in the survey.
span	An integer in c(1,3,5) indicating the span of the desired data.
geography	a valid geo.set object specifying the census geography or geographies to be fetched.

Value

A list containing the endyear, span, a data.frame of estimates, a data.frame of standard errors, a character vector of the original column names, and a data.frame of the geography metadata from acs.fetch.

See Also

acs.fetch, geo.make

rawDR *Raw Death Rate by race and gender*

Description

A dataset containing raw death rate data by race and gender of the deceased. Data is provided for 1980-2013.

Usage

```
rawDR
```

Format

A `data.frame` with 612 observations and 4 variables.

year The year for which data was was recorded.

race The racial group of the deceased One of `all` all races; `white` whites; `black_aa` black / African-American; `nat_amer` American Indian or Native Alaskan; `asian_isl` Asian or Pacific Islander; `hisp_lat` Hispanic.

gender The gender of the deceased. One of `c(both,male,female)`

death_rate The raw death rate. See details.

Details

- The death rate is defined as deaths per 100,000 population.
- Populations are based on census counts enumerated as of April 1 of the census year and estimated as of July 1 for non-census years.

Source

http://www.cdc.gov/nchs/nvss/deaths.htm

References

Xu, J. Q., S. L. Murphy, and K. D. Kochanek. "Deaths: final data for 2013." National Vital Statistics Reports 64.2 (2015).

`split` *Split a "macroACS" object*

Description

Split a "macroACS" object into subsets. This may be helpful for users who have limited memory available on their machines before proceding to derive sample synthetic micro data.

Usage

```
split(acs, n_splits)
```

Arguments

`acs`	An object of class `"macroACS"`.
`n_splits`	An integer for the number of splits you wish to create.

See Also

`derive_synth_datasets`

`stateFR` *Birth rates, by age of mother: United States, each state and territory, 2014*

Description

A dataset containing birth rate data by US state and age for all US states and territories in 2014.

Usage

```
stateFR
```

Format

A `data.frame` with 612 observations and 3 variables.

state The state or territory for which data was was recorded.

age_group The age group of the mother.

birth_rate The birth rate. See Details.

Details

- The birth rate is defined as births per 1,000 women in the specified group.
- Birth rates for `age_group` 45_49 are computed by relating births to women aged 45 and over to women aged 45-49
- Data for the "United States" as a whole excludes data for the territories.
- Data is missing (eg. `NA`) when data does not meet standards of reliability or percision; birth rates based on fewer than 20 births.

Source

http://www.cdc.gov/nchs/nvss/births.htm

References

Hamilton, Brady E., et al. "Births: final data for 2014." National Vital Statistics Reports 64.12 (2015): 1-64.

summary.smsm_set *Summarizing SMSM fits*

Description

summary method for class 'smsm_set'.

Usage

```
## S3 method for class 'smsm_set'
summary(object, ...)
```

Arguments

object	An object of class 'smsm_set', typically a result of call to all_geog_optimize_microdata
...	additional arguments affecting the summary produced.

synthetic_new_attribute *Add a new attribute to a synthetic_micro dataset*

Description

Add a new attribute to a synthetic_micro dataset using conditional relationships between the new attribute and existing attributes (eg. wage rate conditioned on age and education level).

Usage

```
synthetic_new_attribute(df, prob_name = "p", attr_name = "variable",
  conditional_vars = NULL, sym_tbl = NULL)
```

Arguments

df	An R object of class "synthetic_micro".
prob_name	A string specifying the column name of the df containing the probabilities for each synthetic observation.
attr_name	A string specifying the desired name of the new attribute to be added to the data.
conditional_vars	An character vector specifying the existing variables, if any, on which the new attribute (variable) is to be conditioned on. Variables must be specified in order. Defaults to NULL ie- an unconditional new attribute.
sym_tbl	sym_tbl A data.frame symbol table with N + 2 columns. The last two columns must be: 1. A vector containing the new attribute counts or percentages; 2. is a vector of the new attribute levels. The first N columns must match the conditioning scheme imposed by the variables in conditional_vars. See details and examples.

Value

A new synthetic_micro dataset with class "synthetic_micro".

Details

New synthetic variables are introduced to the existing data via conditional probability. Similar to derive_synth_datasets, the goal with this function is to generate a joint probability distribution for an attribute vector; and, to create synthetic individuals from this distribution. Although no limit is placed on the number of variables on which to condition, in practice, data rarely exists which allows more than two or three conditioning variables. Other variables are assumed to be independent from the new attribute.

** There are four different types of conditional/marginal probability models which may be considered for a given new attribute: (1) Independence: it is assumed that each of the variables is independent of the others (2) Pairwise conditional independence: it is assumed that attributes are related to only one other attribute and independent of all others. (3) Conditional independence: Attributes can be depedent on some subset of other attributes and independent of the rest. (4) In the most general case, all attributes are jointly interrelated.

Conditioning is implemented via symbol-tables (sym_tbl) to ensure accurate matching between conditioning variables, new attribute levels, and new attribute probabilities. The symbol table is constructed such that the key in the symbol-table's key-value pair is the specific values for the set of conditioning variables. This key is the first N columns of sym_tbl. A recursive approach is employed to conditionally partition sym_tbl. In this sense, the *order* in which the conditional variables are supplied matters.

The value is final 2 columns of sym_tbl which are a pair of (A) either counts or percentages used to specify the probability for the new attribute and (B) the level that the new attribute takes on.

Examples

```
{
set.seed(567L)
df <- data.frame(gender= factor(sample(c("male", "female"), size= 100, replace= TRUE)),
             edu= factor(sample(c("LT_college", "BA_degree"), size= 100, replace= TRUE)),
                p= runif(100))
df$p <- df$p / sum(df$p)
class(df) <- c("data.frame", "micro_synthetic")
ST <- data.frame(gender= c(rep("male", 3), rep("female", 3)),
                 attr_pct= c(0.1, 0.8, 0.1, 0.05, 0.7, 0.25),
                 levels= rep(c("low", "middle", "high"), 2))
df2 <- synthetic_new_attribute(df, prob_name= "p", attr_name= "SES", conditional_vars= "gender",
         sym_tbl= ST)

ST2 <- data.frame(gender= c(rep("male", 3), rep("female", 6)),
                  edu= c(rep(NA, 3), rep(c("LT_college", "BA_degree"), each= 3)),
                  attr_pct= c(0.1, 0.8, 0.1, 10, 80, 10, 5, 70, 25),
                  levels= rep(c("low", "middle", "high"), 3))
df2 <- synthetic_new_attribute(df, prob_name= "p", attr_name= "SES",
         conditional_vars= c("gender", "edu"),
         sym_tbl= ST2)
}
```

TFR *Total Fertility Rate by race of mother*

Description

A dataset containing total fertility rate data by race of the mother. Data for all races is provided for 1970-2014 and for individual races from 1989-2014.

Usage

```
TFR
```

Format

A `data.frame` with 175 observations and 3 variables.

year The year for which data was was recorded.

race The racial group of the mothers. One of `all` all races; `white` non-hispanic whites; `black_aa` black / African-American; `nat_amer` American Indian or Native Alaskan; `asian_isl` Asian or Pacific Islander; `hisp_lat` Hispanic or Latin American.

tfr The Total Fertility Rate. See Details

Details

The Total Fertility Rate is defined as the sums of the birth rates for the 5-year age groups found in `BR2014` multiplied by 5.

Source

`http://www.cdc.gov/nchs/nvss/births.htm`

References

Hamilton, Brady E., et al. "Births: final data for 2014." National Vital Statistics Reports 64.12 (2015): 1-64.

Index