Package ‘R2BEAT’

January 11, 2021

Type Package

Title Multistage Sampling Allocation and PSU Selection

Description Multivariate optimal allocation for different domains in one and two stages stratified sample design.R2BEAT extends the Neyman (1934) – Tschuprow (1923) allocation method to the case of several variables, adopting a generalization of the Bethel’s proposal (1989). R2BEAT develops this methodology but, moreover, it allows to determine the sample allocation in the multivariate and multi-domains case of estimates for two-stage stratified samples. It also allows to perform Primary Stage Units selection. This package requires the availability of ReGenesees, that can be installed from <https://github.com/DiegoZardetto/ReGenesees>.

Version 1.0.2

Date 2021-01-07

Depends R (>= 3.5.0), plyr, sampling, devtools

NeedsCompilation no

Suggests ReGenesees

Author Andrea Fasulo,Giulio Barcaroli,Raffaella Cianchetta,Stefano Falorsi,Alessio Guandalini,Daniela Pagliuca,Marco Dionisio Terribili

Maintainer Andrea Fasulo <fasulo@istat.it>

License EUPL

Encoding UTF-8

URL https://barcaroli.github.io/R2BEAT/

BugReports https://github.com/barcaroli/R2BEAT/issues

Repository CRAN

Date/Publication 2021-01-11 12:30:05 UTC

R topics documented:

`allocation` *Sample sizes for each stratum*

Description

Example data frame containing a given allocation.

Usage

```
data(beat.example)
```

Format

The Strata data frame contains a row per each stratum with the following variables:

SIZE Stratum sample size (numeric)

Details

Note: the names of the variables must be the ones indicated above.

Examples

```
# Load example data
data(beat.example)
allocation
str(allocation)
```

beat.1st *Compute one stage multivariate optimal allocation.*

Description

Compute multivariate optimal allocation for different domains in one stage stratified sample design

Usage

```
beat.1st(stratif, errors, minnumstrat=2, maxiter=200, maxiter1=25, epsilon=10^(-11))
```

Arguments

`stratif`	Data frame of survey strata, for more details see, e.g.,strata.
`errors`	Data frame of expected coefficients of variation (CV) for each domain, for more details see, e.g.,errors.
`minnumstrat`	Minimum number of elementary units per strata (default=2).
`maxiter`	Maximum number of iterations (default=200) of the general procedure. This kind of iteration may be required by the fact that when in a stratum the number of allocated units is greater or equal to its population, that stratum is set as "census stratum", and the whole procedure is re-initialised.
`maxiter1`	Maximum number of iterations in Chromy algorithm (default=25).
`epsilon`	Tollerance for the maximum absolute differences between the expected CV and the realised CV with the allocation obtained in the last iteraction for all domains. The default is 10^(-11).

Details

The methodology is a generalization of Bethel multivariate allocation (1989) that extended the Neyman (1959) - Tchuprov (1923) allocation for multi-purpose and multi-domains surveys. The generalized Bethel's algorithm allows to determine the optimal sample size for each stratum in a stratified sample design. The overall sample size and the allocation among the different strata is determined starting from the accuracy constraints imposed in the survey on interest estimates.

Value

Object of class `list`. The list contains 4 objects:

`n`	Vector with the optimal sample size for each stratum.
`file_strata`	Data frame corresponding to the input data.frame `stratif` with the `n` optimal sample size column added.
`alloc`	Data frame with optimal (`ALLOC`), proportional (`PROP`), equal (`EQUAL`) sample size allocation.

`sensitivity` Data frame with a summary of expected coefficients of variation (`Planned CV`), realized coefficients of variation with the given optimal allocation (`Actual CV`) and the sensitivity at 10% for each domain and each variable. Sensitivity can be a useful tool to help in finding the best allocation, because it provides a hint of the expected sample size variation for a 10% change in planned CVs.

Author(s)

Developed by Stefano Falorsi, Andrea Fasulo, Alessio Guandalini, Daniela Pagliuca, Marco D. Terribili.

References

Bethel, J. (1989) *Sample allocation in multivariate surveys.* Survey methodology, 15.1: 47-57.

Cochran, W. (1977) *Sampling Techniques.* John Wiley & Sons, Inc., New York

Neyman, J. (1934). On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4): 558-625.

Tschuprow, A. A. (1923). On the mathematical expectation of the moments of frequency distributions in the case of correlated observation. (Chapters 4-6). *Metron*, 2: 646-683.

Examples

```
# Load example data
data(beat.example)

## Example 1
# Allocate the sample
allocation_1 <- beat.1st(stratif=strata, errors=errors)

# The total sample size is
sum(allocation_1$n)

## Example 2
# Assume 5700 units is the maximum sample size to stick to our budget.
# Looking at allocation_1$sensitivity we can see that most of the
# sensitivity is in DOM1 for REG1 and REG2 due to V1.
allocation_1$sensitivity
# We can relax the constraints increasing the expected coefficients of variation for X1 by 10%
errors1 <- errors
errors1[1,2] <- errors[1,2]+errors[1,2]*0.1

# Try the new allocation
allocation_2 <- beat.1st(stratif=strata, errors=errors1)
sum(allocation_2$n)

## Example 3
# On the contrary, if we tighten the constraints decreasing the expected coefficients of variation
# for X1 by 10%
errors2 <- errors
```

```
errors2[1,2] <- errors[1,2]-errors[1,2]*0.1

# The new allocation leads to a larger sample than the first example
allocation_3 <- beat.1st(stratif=strata, errors=errors2)
sum(allocation_3$n)
```

beat.2st *Multivariate optimal allocation for different domains in two stage stratified sample design*

Description

Compute multivariate optimal allocation for different domains corrected considering stratified two stages design

Usage

```
beat.2st(stratif, errors, des_file, psu_file, rho, deft_start = NULL,
    effst = NULL, epsilon1 = 5, mmdiff_deft = 1,maxi = 20,
    epsilon = 10^(-11), minnumstrat = 2, maxiter = 200, maxiter1 = 25)
```

Arguments

stratif	Data frame of survey strata, for more details see, e.g.,strata.
errors	Data frame of coefficients of variation for each domain, for more details see, e.g.,errors.
des_file	Data frame containing information on sampling design variables, for more details see, e.g.,design.
psu_file	Data frame containing information on primary stage units stratification, for more details see, e.g.,PSU_strat.
rho	Data frame of survey strata, for more details see, e.g.,rho.
deft_start	Data frame of survey strata, for taking into account the initial design effect on each variable, for more details see, e.g.,deft_start.
effst	Data frame of survey strata, for taking into account the estimator effect on each variable, for more details see, e.g.,effst.
epsilon1	First stop condition: sample sizes differences beetween two iterations; iteration continues until the maximum of sample sizes differences is greater than the default value. The default is 5.
mmdiff_deft	Second stop condition: defts differences beetween two iterations; iteration continues until the maximum of defts largest differences is greater than the default value. The default is 0.06.
maxi	Third stop condition: maximum number of allowed iterations. The default is 20.
epsilon	The same as in function beat.1st.
minnumstrat	The same as in function beat.1st.
maxiter	The same as in function beat.1st.
maxiter1	The same as in function beat.1st.

Details

The methodology is a generalization of Bethel multivariate allocation (1989) that extended the Neyman (1959) - Tchuprov (1923) allocation for multi-purpose and multi-domains surveys. The generalized Bethel's algorithm allows to determine the optimal sample size for each stratum in a stratified sample design. The overall sample size and the allocation among the different strata is determined starting from the accuracy constraints imposed in the survey on interest estimates. The optimal allocation is obtained throught a procedure that converge in few iteractions:

The first iteration is a computation of an initial allocation with the multivariate optimal allocation for different domains in one stages statified sample design (the methodology is a generalization for multidomains and multistages designs of Bethel multivariate allocation, 1989).

The correction of the initial allocation is based on an iterative method calculating new allocations and is based on an inflaction of strata variances using the design effect (Ganninger, 2010).

Value

Object of class `list`. The list contains 8 objects:

`iteractions` Data frame that for each iteraction provides a summary with the number of Primary Stage Units (`PSU_Total`) distinguish between Self-Representative (`PSU_SR`) from Non-Self-Representative (`PSU_NSR`) and the number of Secondary Stage Units (`SSU`). This output is also printed to the screen.

`file_strata` Input data frame in `stratif` with the design effect for each variables in each stratum (`DEFT1 -DEFTn`) and the optimal sample size columns.

`alloc` Data frame with optimal (`ALLOC`), proportional (`PROP`), equal (`EQUAL`) sample size allocation.

`planned` Data frame with a summary of expected coefficients of variation for each variable in each domain.

`expected` Data frame with a summary of realized coefficients of variation with the given optimal allocation for each variable in each domain.

`sensitivity` Data frame with a summary of the sensitivity at 10% for each domain and each variable. Sensitivity can be a useful tool to help in finding the best allocation, because it provides a hint of the expected sample size variation for a 10% change in planned CVs.

`deft_c` Data frame with the design effect for each variable in each domain in each iteraction. Note that `DEFT1_0 -DEFTn_0` is always equal to 1 if `deft_start` is `NULL`. Instead is equal to `deft_start`. While `DEFT1 -DEFTn` are the final design effect related to the given allocation.

`param_alloc` A vector with a resume of all the parameter given for the allocation.

Author(s)

Developed by Stefano Falorsi, Andrea Fasulo, Alessio Guandalini, Daniela Pagliuca, Marco D. Terribili.

References

Cochran, W. (1977) *Sampling Techniques.* John Wiley & Sons, Inc., New York

Ganninger, M. (2010). *Design effects: model-based versus design-based approach.* Vol. 3, p. 174. DEU.

Neyman, J. (1934). On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4), 558-625.

Tschuprow, A. A. (1923). On the mathematical expectation of the moments of frequency distributions in the case of correlated observation. (Chapters 4-6). *Metron*, 1923, 2: 646-683.

Examples

```
# Load example data
data(beat.example)

## Example 1
# Allocate the sample
allocation2st_1 <- beat.2st(stratif=strata, errors=errors,
des_file=design, psu_file=PSU_strat,rho=rho)
# The total ammount of sample size is 191 PSU (36 SR + 155 NSR) and 15147 SSU.

## Example 2
# Assume 13000 SSUs is the maximum sample size to stick to our budget.
# Look at the sensitivity is in DOM1 for REG1 and REG2 due to V1.
allocation2st_1$sensitivity
# We can relax the constraints increasing the expected coefficients of variation for X1 by 10%
errors1 <- errors
errors1[1,2] <- errors[1,2]+errors[1,2]*0.1

# Try the new allocation
allocation2st_2 <- beat.2st(stratif=strata, errors=errors1,
des_file=design, psu_file=PSU_strat,rho=rho)

## Example 3
# On the contrary, if we tighten the constraints decreasing the expected coefficients of variation
# for X1 by 10%
errors2 <- errors
errors2[1,2] <- errors[1,2]-errors[1,2]*0.1

# The new allocation leads to a larger sample than the first example (around 18000)
allocation2st_3 <- beat.2st(stratif=strata, errors=errors2,
des_file=design, psu_file=PSU_strat,rho=rho)

## Example 4
# Sometimes some budget constraints concern the number of PSU involved in the survey.
# Tuning the PSUs number is possible modyfing the MINIMUM in des_file.
# Assume to increase the MINIMUM from 48 to 60
design1 <- design
design1[,4] <- 60
allocation2st_4 <- beat.2st(stratif=strata, errors=errors2,
des_file=design1, psu_file=PSU_strat, rho=rho)

# The PSUs numer is decreased, while the SSUs number increased
# due to cluster intra-correlation effect.
# Under the same expected errors, to offset a slight reduction of PSUs (from 221 to 207)
# an increase of SSUs involved is observed.
allocation2st_3$expected
allocation2st_4$expected

## Example 5
# On the contrary, assume to decrease the MINIMUM from 48 to 24.
# The SSUs number strongly decrease in the face of an increase of PSUs,
# always under the same expected errors.
design2 <- design
design2[,4] <- 24
allocation2st_5 <- beat.2st(stratif=strata, errors=errors2,
des_file=design2, psu_file=PSU_strat, rho=rho)
allocation2st_4$expected
allocation2st_5$expected

# Note that MINIMUM can be different for each stratum.

## Example 6
# Assume that the SSUs are in turn clusters, for instance households composed by individuals.
# In the previous examples we always derived optimal allocations
# for sample of SSUs (i.e. households, because
# DELTA = 1).
design
design1
design2
# For obtaining a sample in terms of the elements composing SSUs
# (i.e., individuals) is just sufficient to
# modify the DELTA in des_file.
design3 <- design
design3$DELTA <- 2.31
# DELTA_IND=2.31, the average size of household in Italy.
allocation2st_6 <- beat.2st(stratif=strata, errors=errors,
des_file=design3, psu_file=PSU_strat, rho=rho)
```

beat.cv *Computation of coefficient of variation (CV) for a given multivariate multiple allocation*

Description

Compute the coefficients of variation considering a given multivariate optimal allocation.

Usage

```
beat.cv(n_file, stratif, errors, des_file, psu_file, rho, epsilon)
```

Arguments

`n_file` Data frame containing the sample size allocated in each stratum, for more details, e.g.,allocation.

`stratif` Data frame of survey strata, for more details see, e.g.,strata.

`errors` Data frame of expected coefficients of variation (CV) for each domain, for more details see, e.g.,errors.

`des_file` Data frame containing information on sampling design variables, for more details see, e.g.,design.

`psu_file` Data frame containing information on primary stage units stratification, for more details see, e.g.,PSU_strat.

`rho` Data frame of survey strata, for more details see, e.g.,rho.

`epsilon` The same as in function beat.1st.

Details

This function enables to derive the expected coefficient of variation (CV) from a given allocation. The function `beat.cv` returns the estimates expected accuracy in terms of coefficient of variation, for several variables in different domains, given a certain allocation among the different strata.

Value

Object of class `list`. The list contains a set of `data.frame`, as many of the cross product between domain and interest variables, containing total estimates, population, variance and expected coefficient of variation for every domain modality. For each domain and each variable is defined

`Tot1` Total estimate

`N` Measure of size

`Varfin` The sample variance of the total estimate

`CV` The coefficient of variation of the

Author(s)

Developed by Stefano Falorsi, Andrea Fasulo, Alessio Guandalini, Daniela Pagliuca, Marco D. Terribili.

Examples

```
# Load example data
data(beat.example)

## Example 1
# Calculate coefficients of variation, for two variables in two domains,
# given an allocation among the different strata.
allocation
cv1<-beat.cv( n_file=allocation, stratif=strata, errors=errors,
des_file=design, psu_file=PSU_strat, rho=rho)
```

```
## Example 2
# Take the example 1 in beat.2st.
allocation2st_1 <- beat.2st(stratif=strata, errors=errors,
des_file=design, psu_file=PSU_strat,rho=rho)
# The allocation obtained is
allocation2st_1$alloc
# with these precision constraints
errors
# and these expected coefficient of variation
allocation2st_1$expected

# Now, fit the output of beat.2st to allocation, that is
SIZE <- allocation2st_1$alloc[-18,c(2)]
allocation1 <- data.frame(SIZE)
# If apply beat.cv the same error in allocation2st_1$expected should be obtained.
# In fact
cv2<-beat.cv( n_file=allocation1, stratif=strata, errors=errors,
des_file=design, psu_file=PSU_strat, rho=rho)
cv2

# Please, note that some very slightly differences may occur.
```

check_input *Check of coherence in the inputs for the allocation step*

Description

Checks the coherence between the population in the strata dataset and the population calculated by the PSUs dataset

Usage

```
check_input(strata,des,strata_var_strata,strata_var_des)
```

Arguments

strata	strata dataset
des	design dataset
strata_var_strata	variable identifying stratum in strata dataset
strata_var_des	variable identifying stratum in design dataset

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(R2BEAT)

load(”R2BEAT_ReGenesees.RData”)  # ReGenesees design and calibration objects plus PSU data

RGdes <- des                           # ReGenesees design object
RGcal <- cal                           # ReGenesees calibrated object

strata_vars <- c(”stratum”)            # variables of stratification
target_vars <- c(”income_hh”,
                 ”active”,
                 ”inactive”,
                 ”unemployed”)         # target variables
deff_vars <- ”stratum”    # stratification variables for calculating deff and effst
                      # (n.b: must coincide or be a subset of variables of stratification)
id_PSU <- c(”municipality”)            # identification variable of PSUs
id_SSU <- c(”id_hh”)                   # identification variable of SSUs
domain_vars <- c(”region”)             # domain variables
inp1 <- check_input(RGdes,
                            RGcal,
                            id_PSU,
                            id_SSU,
                            strata_vars,
                            target_vars,
                            deff_vars,
                            domain_vars)
head(inp1$strata)
head(psu)
psu_id=”municipality”        # Identifier of the PSU
stratum_var=”stratum”        # Identifier of the stratum
mos_var=”ind”                # Variable to be used as 'measure of size'
delta=1                      # Average number of SSUs for each selection unit
minimum <- 50                # Minimum number of SSUs to be selected in each PSU
inp2 <- input_to_beat.2st_2(psu,
                            psu_id,
                            stratum_var,
                            mos_var,
                            delta,
                            minimum)
head(inp2$psu_file)
head(inp2$des_file)
newstrata <- check_input(strata=inp1$strata,
                         des=inp2$des_file,
                         strata_var_strata=”STRATUM”,
                         strata_var_des=”STRATUM”)

## End(Not run)
```

`deft_start` *Starting values for the Design Effect ($deft$)*

Description

Example data frame containing the starting values for the Design Effect ($deft$) .

Usage

```
data(beat.example)
```

Format

The Design Effect data frame contains a row per each stratum with the following variables:

STRATUM Identifier of the stratum (numeric).

DEFT1 Starting values for the Design Effect in the stratum of the first variable (numeric).

DEFTj Starting values for the Design Effect in the stratum of the j-th variable (numeric).

DEFTn Starting values for the Design Effect in the stratum of the last variable (numeric).

Details

Note: the names of the variables must be the ones indicated above.

This is an optional input. The function `beat.2st` independently computes and updates the design effect. However, it is possible to set the starting values of design effect for each variable in each stratum. The design effect is the square root of the ratio of the actual sampling variance to the variance expected with the simple random sampling (SRS), on equal sample size.

Under SRS the desing effect is equal to 1. Usually, as increasing the stages of selection the design effect increases because it takes into account the "clusterization" of sampling units and the sample size in Self Representative (SR) and Non Self Representative (NSR) strata.

In practice, higher is the intraclass correlation, higher will be the design effect and much more sample size for satisfying the precision constraints is needed with respect to SRS.

Examples

```
# Load example data
data(beat.example)
deft_start
str(deft_start)
```

`design` *Sampling design variables*

Description

Example data frame containing variables for describing the sampling design.

Usage

```
data(beat.example)
```

Format

The design data frame contains a row per each stratum with the following variables:

STRATUM Identifier of the stratum (numeric)

STRAT_MOS Measure of size of the stratum (numeric)

DELTA The average size of Secondary Stage Units (SSU) in the strata. With respect to the sample on which we are interested in, it could be equal or greater than 1 (numeric). See details for a depth explanation.

MINIMUM the minimum number of SSU to be selected in each PSU. It could be different in each stratum (numeric)

Details

Note: the names of the variables must be the ones indicated above.

The sample design can be defined through a measure of size of the stratum, the average size of each SSU (>=1) and the minimum number of SSU to be selected in each PSU. In particular, if SSU are not cluster DELTA=1 and the sample size determined will be given in term of SSU. Instead, when SSUs are, in turn, clusters (for instance, households composed by individuals), defining DELTA equal to the average size of SSUs, enables to derive a sample in term of individuals.

Furthermore, modifying the MINIMUM it is possible to tune the number of PSU in the sample (see the example in beat.1st). In fact, considering the same sample size, increasing the MINIMUM, less PSU will be involved in the sample, but worst estimates in term of expected coefficient of variations will be provided. On the contrary, decreasing the MINIMUM, more PSU will be involved in the sample and better estimates will be obtained. Instead, increasing the MINIMUM for obtaining the same expected errors, requests less PSU, but much more SSU. The contrary occurs decreasing the MINIMUM.

Examples

```
# Load example data
data(beat.example)
design
str(design)
```

---

effst *Estimator effect*

Description

Example data frame containing estimator effect, $(effst)$, in each stratum for each variable.

Usage

```
data(beat.example)
```

Format

The estimator effect data frame contains a row per each stratum with the following variables:

STRATUM Identifier of the stratum (numeric).

EFFST1 Estimator effect in the stratum of the first variable (numeric).

EFFSTj Estimator effect in the stratum of the j-th variable (numeric).

EFFSTn Estimator effect in the stratum of the last variable (numeric).

Details

Note: the names of the variables must be the ones indicated above.

The estimator effect, ($effst$), provides a measure of the variance inflaction or reduction due to the use of a different estimator from the HT (Horvitz and Thompson, 1952). It is equal to the ratio between the sampling variance of the estimator planned to be used and the sampling variance of the HT.

Then, when the HT is used, ($effst$) is equal to 1. However, always more often, different estimators, such as calibration estimator (Deville and Särndal, 1992) or generalized regression estimator GREG (Fuller, 2002 and references therein), are used. Usually this kind of estimators take into account auxiliary variables that enables to increase the accuracy of the estimates, that is, they reduce their errors (CV). Then, their $effst$ is usually lower than 1.

Therefore, taking into account the estimator effect when planning the survey can help in saving sample size or at least to more properly evatuate the allocation.

References

Deville, J.C., Särndal, C.E. (1992). Calibration estimators in survey sampling. *Journal of the American statistical Association*, 87(418): 376-38.

Fuller, W.A.. (2002). Regression estimation for survey samples. Survey Methodology 28(1): 5-23.

Horvitz, D.G., Thompson, D.J. (1952) A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260): 663-685.

Examples

```
# Load example data
data(beat.example)
effst
str(effst)
```

errors *Precision constraints (maximum CVs) as input for Bethel allocation*

Description

Example data frame containing precision levels (expressed in terms of acceptable CV's).

Usage

```
data(beat.example)
```

Format

The constraint data frame (errors) contains a row per each type of domain with the following variables:

DOM Type of domain code (factor).

CV1 Planned coefficient of variation for first variable (numeric).

CVj Planned coefficient of variation for j-th variable (numeric).

CVn Planned coefficient of variation for last variable (numeric).

Details

Note: the names of the variables must be the ones indicated above.

The coefficient of variation (CV) is a standardized measure of variance. It is often expressed as a percentage and is defined as the ratio between the standard deviation of the estimate and the estimate (or its absolute value).

Examples

```
# Load example data
data(beat.example)
errors
str(errors)
```

input_to_beat.2st_1 *Input dataframes for R2BEAT two-stages sample design*

Description

Prepares the following input dataframes for R2BEAT two-stages sample design starting from ReGenesees design and/or calibrated objects: 1. strata 2. deff 3. effst 4. rho

Usage

```
input_to_beat.2st_1(RGdes,
                    RGcal,
                    id_PSU,
                    id_SSU,
                    strata_vars,
                    target_vars,
                    deff_vars,
                    domain_vars)
```

Arguments

`RGdes`	ReGenesees design object.
`RGcal`	ReGenesees calibrated object.
`id_PSU`	variables used as identifiers in ReGenesees objects.
`id_SSU`	variables used as identifiers in ReGenesees objects.
`strata_vars`	stratification variables used in ReGenesees objects.
`target_vars`	target variables.
`deff_vars`	stratification variables to be used when calculating deff.
`domain_vars`	the variables used to identify the domain of interest.

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(R2BEAT)

load("R2BEAT_ReGenesees.RData")  # ReGenesees design and calibration objects plus PSU data

RGdes <- des                           # ReGenesees design object
RGcal <- cal                           # ReGenesees calibrated object

strata_vars <- c("stratum")            # variables of stratification
target_vars <- c("income_hh",
                 "active",
                 "inactive",
                 "unemployed")         # target variables
deff_vars <- "stratum"    # stratification variables for calculating deff and effst
                      # (n.b: must coincide or be a subset of variables of stratification)
id_PSU <- c("municipality")            # identification variable of PSUs
id_SSU <- c("id_hh")                   # identification variable of SSUs
domain_vars <- c("region")             # domain variables
inp1 <- input_to_beat.2st_1(RGdes,
                            RGcal,
                            id_PSU,
                             id_SSU,
                             strata_vars,
                             target_vars,
                             deff_vars,
                             domain_vars)
inp1$strata
inp1$deff
inp1$effst
inp1$rho

## End(Not run)
```

---

input_to_beat.2st_2 *Prepares the design file for two-stage sample design*

Description

Prepares the design file for two-stage sample design on the basis of a dataset containin information on each PSU

Usage

```
input_to_beat.2st_2(psu,psu_id,stratum_var,mos_var,delta,minimum)
```

Arguments

psu	Dataframe containing information on each PSU.
psu_id	Identifier of each PSU in PSU dataframe.
stratum_var	Identifier of stratum in PSU dataframe.
mos_var	Variable containing the number of selection units in each PSU.
delta	Average number of final number of SSU per each selection unit.
minimum	Minimum number of selection units to be interviewed in each PSU.

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
# psu <- read.csv2("psu.csv") # Read the external file containing PSU information
load("R2BEAT_ReGenesees.RData") # ReGenesees design and calibration objects plus PSU data
head(psu)
psu_id="municipality"        # Identifier of the PSU
stratum_var="stratum"        # Identifier of the stratum
mos_var="ind"                # Variable to be used as 'measure of size'
delta=1                      # Average number of SSUs for each selection unit
minimum <- 50                # Minimum number of SSUs to be selected in each PSU
```

```
inp2 <- input_to_beat.2st_2(psu,
                            psu_id,
                            stratum_var,
                            mos_var,
                            delta,
                            minimum)
head(inp2$psu_file)
head(inp2$des_file)

## End(Not run)
```

PSU_strat *Information on Primary Stage Units (PSUs) stratification*

Description

Example data frame containing information on Primary Stage Units (PSUs) stratification.

Usage

```
data(beat.example)
```

Format

The PSU_strat data frame contains a row for each Primary Stage Units (PSUs) with the following variables:

STRATUM Identifier of the stratum (numeric)

PSU_MOS Measure of size of the primary stage unit (numeric)

PSU_ID Identifier of the primary stage unit (numeric)

Details

Note: the names of the variables must be the ones indicated above.

Examples

```
# Load example data
data(beat.example)
PSU_strat
str(PSU_strat)
```

rho *Intraclass correlation coefficients for self and non self representative in the strata*

Description

Example data frame containing intraclass correlation ρ in Self Representative (SR) and Non Self Representative (NSR) strata.

Usage

```
data(beat.example)
```

Format

The intraclass correlation coefficienta (ρ) data frame contains a row per each stratum with the following variables:

STRATUM Identifier of the stratum (numeric)

RHO_AR1 intraclass correlation of the elementary units for each primary stage unit of the self representing area belonging to the stratum for the first variable.

RHO_ARj intraclass correlation of the elementary units for each primary stage unit of the self representing area belonging to the stratum for the j-th variable.

RHO_ARn intraclass correlation of the elementary units for each primary stage unit of the self representing area belonging to the stratum for the n-th variable.

RHO_NAR1 intraclass correlation of the elementary units for each primary stage unit of the non self representing area belonging to the stratum for the first variable.

RHO_NARj intraclass correlation of the elementary units for each primary stage unit of the non self representing area belonging to the stratum for the j-th variable.

RHO_NARn intraclass correlation of the elementary units for each primary stage unit of the non self representing area belonging to the stratum for the n-th variable.

Details

Note: the names of the variables must be the ones indicated above.

Intraclass correlation, ρ, provide a measure of the cluster heterogeneity and they have a direct impact on the design effect (design). It can be indirectly computed from the design effect and the average minimum number of interviews in the Primary Stage Units (PSUs).

The ideal situation is when all the clusters in which the population is divided are more heterogeneous possible within them. At the limit, if each cluster were a reduced copy of the population then it would be sufficient to extract one just to have the same information that would be obtained from a complete survey. Then, more similar the units in the cluster are, higher the sample size must be (Cochran, 1977, Chapter 8).

By definition, in SR strata ρ, is equal to 1, because there is just a single PSU in SR strata. In NSR strata usually, ρ is usual higher than 1, because a double stage of selection is needed.

References

Cochran, W. (1977) *Sampling Techniques.* John Wiley & Sons, Inc., New York.

Examples

```
# Load example data
data(beat.example)
rho
str(rho)
```

---

strata *Strata characteristics*

Description

Example data frame containing information on strata characteristics.

Usage

```
data(beat.example)
```

Format

The Strata data frame contains a row per each stratum with the following variables:

STRATUM Identifier of the stratum (numeric).

N Stratum population size (numeric).

M1 Mean in the stratum of the first variable (numeric).

Mj Mean in the stratum of the j-th variable (numeric).

Mn Mean in the stratum of the last variable (numeric).

S1 Standard deviation in the stratum of the first variable (numeric).

Sj Standard deviation in the stratum of the j-th variable (numeric).

Sn Standard deviation in the stratum of the last variable (numeric).

CENS flag (1 indicates a take all stratum, 0 a sampling stratum, usually 0) (numeric).

COST Cost per interview in each stratum, usually 0 (numeric).

DOM1 Domain value to which the stratum belongs for the first type of domain (factor or numeric).

DOMa Domain value to which the stratum belongs for the a-th type of domain (factor or numeric).

DOMk Domain value to which the stratum belongs for the k-th type of domain (factor or numeric).

Details

Note: the names of the variables must be the ones indicated above.

Examples

```
# Load example data
data(beat.example)
strata
str(strata)
```

StratSel *Stratification and selection*

Description

Merge two data frames (whatever PSU population and allocation data frames) and then compute stratification and selection of a fixed number of sample PSUs (Primary Sampling Units) for stratum using the Sampford's method (unequal probabilities, without replacement, fixed sample size), implemented by the UPsampford function of R package *sampling*. About the stratification, the function realizes for each domain estimation, the computation of a size threshold for a given size PSU. PSUs with measure of size that exceeds a calculated threshold are identify like SR (Self Representative) and each constitutes a stratum by itself, so they come into the sample with probability equal to one. The remaining NSR (Non Self Representative) PSUs are ordered for the measure of size and divided into stratum having size approximately constant to the corrected threshold and with PSUs having sizes as homogeneous as possible.
If the number of NSR PSUs in each stratum is greater than the number of sample PSUs for each stratum, then it is indispensable to define the vector of the inclusion probabilities (pik), argument of UPsampford function, vice versa NSR PSUs will become SR PSUs. If some pik values are greater than 1, pik is newly calculated, for those PSUs belonging to the same domain and stratum of the PSUs having pik >1, until pik values will be all less than 1, then applies the UPsampford function.

Usage

```
StratSel(dataPop, idpsu, dom, final_pop, size, PSUsamplestratum, min_sample,
min_sample_index = FALSE, dataAll, domAll, f_sample,
planned_min_sample = NULL, launch = TRUE)
```

Arguments

dataPop	PSU population data frame.
idpsu	Formula identifying primary sample units identifier.
dom	Formula identifying the variable domain.
final_pop	Formula identifying secondary stage units.
size	Formula identifying first stage units size.
PSUsamplestratum	Number of sample PSUs to select to each stratum.
min_sample	Number of final sample units to observe in each PSU.
min_sample_index	Identify the absence of planned_min_sample (default = FALSE)

`dataAll` Allocation data frame.

`domAll` Formula identifying the domain variable for allocation data frame.

`f_sample` Formula identifying final sample units.

`planned_min_sample`
Formula identifying planned final sample units to observe in each PSU, variable between domains; `NULL` (the default) means the existence of a fixed number of them identify from min_sample.

`launch` Identify the parameter related to the launch procedure. If default = `TRUE` the launch is partial (see 'Details').

Details

It is possible to launch the procedure in two separate steps, so that user can see a first output (launch = `TRUE`) and decide if modify the input parameters or continue the procedure setting launch = `FALSE`.

Value

An object of class `list` or `data.frame` depending from argument `launch`. If launch = `TRUE` the only output is, at domain level, a data frame of the Self Representative (SR) PSUs and of the Non Self Representative (NSR) PSUs before stratification. If launch = `FALSE` the output is a `list` composed by four members of `data.frame` class. The first component, at domain level, is the same data frame obtained if launch = `TRUE`. The second element is, at domain level, a data frame of SR and NSR PSUs after stratification, with their totals like the first output, but with additional information such as the final units sample size distinctly for SR and NSR PSUs and their mean. The third component is a data frame that supply, for each stratum, the number of sample PSUs selected and the total number of PSUs. The fourth element provides, for each PSU, some information like the inclusion probability and the sampling fraction.

Author(s)

Raffaella Cianchetta

References

S. Falorsi A. Russo (2001), Il disegno di rilevazione per indagini Panel sulle famiglie, *Rivista di Statistica Ufficiale, N. 3*, p. 55-90.
Sampford, M. (1967), On sampling without replacement with unequal probabilities of selection, *Biometrika*, 54:499-513.
Yves Tille' and Alina Matei (2012). sampling: Survey Sampling. R package version 2.5.
http://CRAN.R-project.org/package=sampling.

Examples

```
# Start StratSel
## Not run:
load(FS4)
data(population)
data(allocation)
```

```
# The function with a fixed number of final sample units (min_sample= 8)
# to observe in each PSU and partial launch of the procedure with
# only one output data frame(see 'Value')
Output_list <- StratSel(dataPop= population, idpsu= ~ comune_num, dom= ~ dom,
               final_pop= ~ fam, size= ~ pop, PSUsamplestratum= 6, min_sample= 8,
               min_sample_index= FALSE, dataAll= allocation, domAll= ~ dom,
               f_sample= ~ campdom, planned_min_sample= NULL, launch= TRUE)

# It can also be written as below due to default values:
Output_list <- StratSel(dataPop= population, idpsu= ~ comune_num, dom= ~ dom,
               final_pop= ~ fam, size= ~ pop, PSUsamplestratum= 6, min_sample= 8,
               dataAll= allocation, domAll= ~ dom, f_sample= ~ campdom)


# The function with a fixed number of final sample units (min_sample= 8)
# to observe in each PSU and full launch of the procedure with
# the output list composed of four data frames(see 'Value')
Output_list <- StratSel(dataPop= population, idpsu= ~ comune_num, dom= ~ dom,
               final_pop= ~ fam,  size= ~ pop, PSUsamplestratum= 6, min_sample= 8,
               dataAll= allocation, domAll= ~ dom, f_sample= ~ campdom, launch= FALSE)


# The function with a variable number of final sample units (planned_min_sample=
# ~ planned_final_sample) and partial launch of the procedure
Output_list <- StratSel(dataPop= population, idpsu= ~ comune_num, dom= ~ dom,
               final_pop= ~ fam, size= ~ pop, PSUsamplestratum= 6, min_sample= NULL,
               min_sample_index= TRUE, dataAll= allocation, domAll= ~ dom,
               f_sample= ~ campdom, planned_min_sample= ~ planned_final_sample)


# The function with a variable number of final sample units (planned_min_sample=
# ~ planned_final_sample) and full launch of the procedure with the output list
#composed of four data frames
Output_list <- StratSel(dataPop= population, idpsu= ~ comune_num, dom= ~ dom,
               final_pop= ~ fam, size= ~ pop, PSUsamplestratum= 6, min_sample= NULL,
               min_sample_index= TRUE, dataAll= allocation, domAll= ~ dom,
               f_sample= ~ campdom, planned_min_sample= ~ planned_final_sample,
               launch= FALSE)

## End(Not run)
```

Index