Package 'R2BEAT'

May 25, 2023

Type Package

Title Multistage Sampling Allocation and Sample 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 both Primary Stage Units and Secondary Stage Units selection. This package requires the availability of 'ReGenesees', that can be installed from <https://github.com/DiegoZardetto/ReGenesees>.

Version 1.0.5

Depends R (>= 3.5.0), sampling, glue, methods, parallel, foreach, doParallel

Suggests ReGenesees

Author Andrea Fasulo,Giulio Barcaroli,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

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2023-05-25 10:20:06 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), minPSUstrat = 2, 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.
minPSUstrat	Minimum number of non-self-represenative PSUs to be selected in each stratum.
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

```
## Not run:
# 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 number 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


## 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)

## Example 7
# Complete workflow
```

```
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.06),
                         CV3=c(0.03,0.06),
                         CV4=c(0.05,0.08)))
cv
samp_frame <- pop
samp_frame$one <- 1
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")
deff_var <- "stratum"
domain_var <- "region"
delta =  1       # households = survey units
minimum <- 50    # minimum number of SSUs to be interviewed in each selected PSU
deff_sugg <- 1.5 # suggestion for the deff value

inp <- prepareInputToAllocation1(samp_frame,
                                 id_PSU,
                                 id_SSU,
                                 strata_var,
                                 target_vars,
                                 deff_var,
                                 domain_var,
                                 minimum,
                                 delta,
                                 deff_sugg)
inp$desfile$MINIMUM <- 50
alloc <- beat.2st(stratif = inp$strata,
                  errors = cv,
                  des_file = inp$des_file,
                  psu_file = inp$psu_file,
                  rho = inp$rho,
                  deft_start = NULL,
                  effst = inp$effst,
                  minPSUstrat = 2,
                  minnumstrat = 50
)


## End(Not run)
```

`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

```
## Not run:

# 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.

## End(Not run)
```

---

build_dummy_variables *Derives dummy variables from target variables to the sampling frame*

Description

This function allow to derive dummy variables to the sampling frame (this is necessary if we want to differentiate precision constraints in the same domain level, for instance in the different regions).

Usage

```
build_dummy_variables(frame,
                      domain_var,
                      initial_target_vars,
                      cv)
```

Arguments

frame	The sampling frame.
domain_var	In the sampling frame, the variable that identifies a given domain level.
initial_target_vars	In the sampling frame, the initial set of target variables (non dummies)
cv	The initial set of precision constraints, related to the initial set of target variables.

Value

A list containing (i) frame: the sampling frame with the derived dummy variables, (ii) new_target_variables: a vector with the names of the total set of target variables (initial ones plus derived variables), (iii) cvnew: the dataframe containing the new set of precision constraints (related to the initial target variables plus the derived dummy variables)

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
load("frame_pop_EA.RData")
pop <- frame_pop_EA
cv <- as.data.frame(list(DOM = c("DOM1","DOM2"),
                         CV1 = c(0.10,0.20)))
cv
new <- build_dummy_variables(frame = pop,
                             domain_var = "region",
                             initial_target_vars = "unemployed",
                             cv = cv)
head(new$frame)
new$new_target_vars
new$cvnew

## End(Not run)
```

---

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 <- input_to_beat.2st_1(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

```
## Not run:
# Load example data
data(beat.example)
deft_start
str(deft_start)

## End(Not run)
```

`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

```
## Not run:
# Load example data
data(beat.example)
design
str(design)

## End(Not run)
```

`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

```
## Not run:
# Load example data
data(beat.example)
effst
str(effst)

## End(Not run)
```

`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

```
## Not run:
# Load example data
data(beat.example)
errors
str(errors)

## End(Not run)
```

`eval_2stage` *Evaluation of the two-stage sample design optimized solution by simulation*

Description

The user can indicate the number of samples that must be selected by the optimized frame. First, the true values of the parameters are calculated from the frame. Then, for each sample the sampling estimates are calculated, together with the differences between them and the true values of the parameters. At the end, an estimate of the CV is produced for each target variable, in order to compare them with the precision constraints set at the beginning of the optimization process. If the flag 'writeFiles' is set to TRUE, boxplots of distribution of the CV's in the different domains are produced for each Y variable ('cv.pdf'), together with boxplot of the distributions of differences between estimates and values of the parameters in the population ('differences.pdf').

Usage

```
eval_2stage(df,
            PSU_code,
            SSU_code,
            domain_var,
            target_vars,
            PSU_sampled,
            nsampl = 100,
            writeFiles = TRUE)
```

Arguments

`df`	The sampling frame.
`PSU_code`	In the sampling frame, the identifier of the PSU.
`SSU_code`	In the sampling frame, the identifier of the SSU.
`domain_var`	In the sampling frame, the identifier of the domain of interest for the estimates.
`target_vars`	In the sampling frame, the variables used to produce the target estimates.
`PSU_sampled`	The set of selected PSUs.
`nsampl`	The number of samples to be drawn from the frame.
`writeFiles`	A flag to write in the work directory the outputs of the function. Default is TRUE.

Value

A list containing (i) the CV distribution in the domains, (ii) the bias distribution in the domains, (iii) the dataframe containing the sampling estimates by domain

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.05),
                         CV3=c(0.03,0.05),
                         CV4=c(0.05,0.08)))
samp_frame <- pop
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")   # more than one
deff_var <- "stratum"
domain_var <- "region"
delta =  1     # households = survey units
minimum <- 50  # minimum number of SSUs to be interviewed in each selected PSU
deff_sugg <- 1.5  # suggestion for the deff value
inp <- prepareInputToAllocation1(samp_frame,
                                 id_PSU,
                                 id_SSU,
                                 strata_var,
                                 target_vars,
                                 deff_var,
                                 domain_var,
                                 minimum,
                                 delta,
                                 deff_sugg)
inp$desfile$MINIMUM <- 50
alloc <- beat.2st(stratif = inp$strata,
                  errors = cv,
                  des_file = inp$des_file,
                  psu_file = inp$psu_file,
                  rho = inp$rho,
                  deft_start = NULL,
                  effst = inp$effst,
                  minPSUstrat = 2,
                  minnumstrat = 50
                  )
sample_1st <- select_PSU(alloc, type="ALLOC", pps=TRUE)
```

```
df=pop
df$one <- 1
PSU_code="municipality"
SSU_code="id_ind"
target_vars <- c("income_hh",
                 "active",
                 "inactive",
                 "unemployed")

PSU_sampled <- sample_1st$sample_PSU
eval <- eval_2stage(df,
                    PSU_code,
                    SSU_code,
                    domain_var,
                    target_vars,
                    PSU_sampled,
                    nsampl=10,
                    writeFiles=TRUE)
eval$coeff_var
eval$rel_bias

## End(Not run)
```

`input_to_beat.2st_1` *Input dataframes for R2BEAT two-stages sample design (when a previous round of the survey is available, but no sampling frame)*

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(readr)
des <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/des.rds?raw=true")
cal <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/cal.rds?raw=true")

library(R2BEAT)
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 and psu file for two-stage sample design (when a previous round of the survey is available, but no sampling frame)*

Description

Prepares the design file for two-stage sample design on the basis of a dataset containing 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:
library(readr)
psu <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/psu.rds?raw=true")
head(psu)
library(R2BEAT)
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)
```

plot_sens *Plot of the sensitivity analysis for some parameters by means of grid search*

Description

This function allows to plot the results of the simulation carried out by using the function 'sensitivity'.

Usage

```
plot_sens (
  x,
  min,
  max)
```

Arguments

x	The result of the 'sensitivity' function.
min	minimum value of the parameter.
max	maximum value of the parameter.

Value

A list containing the (i) vector of allocated PSUs in the iterations and (ii) the vector of allocated SSUs in the iterations

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.05),
                         CV3=c(0.03,0.05),
                         CV4=c(0.04,0.08)))
cv
# parameters
samp_frame <- pop
errors <- cv
id_PSU <- "municipality"
id_SSU <- "id_ind"
```

```
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")   # more than one
deff_var <- "stratum"
domain_var <- "region"
minimum <- 50 # minimum number of SSUs to be interviewed in each selected PSU
# average dimension of the SSU in terms of elementary survey units
#delta =  nrow(pop) /length(unique(pop$id_hh))
delta =  1  # average dimension of the SSU in terms of elementary survey units
deff_sugg <- 1.5
min <- 30
max <- 80
sens <- sensitivity_min_SSU (
  samp_frame,
  errors,
  id_PSU,
  id_SSU,
  strata_var,
  target_vars,
  deff_var,
  domain_var,
  minimum,
  delta,
  deff_sugg,
  min,
  max)
plot_sens(sens,min,max)

## End(Not run)
```

`prepareInputToAllocation1` *Input dataframes for R2BEAT two-stages sample design when sampling frame is available*

Description

In case of scenario 1 (no previous round of the survey available), prepares the following input dataframes for R2BEAT two-stages sample design starting from the sampling frame: 1. strata 2. deff 3. effst 4. rho 5. PSU_file 6. des_file

Usage

```
prepareInputToAllocation1 (
  samp_frame,
  id_PSU,
  id_SSU,
  strata_var,
  target_vars,
  deff_var,
```

```
  domain_var,
  minimum,
  delta,
  deff_sugg)
```

Arguments

samp_frame	The dataframe containing sampling units in the reference population.
id_PSU	variables used as identifiers in sampling frame.
id_SSU	variables used as identifiers in sampling frame.
strata_var	stratification variable used in sampling frame.
target_vars	target variables.
deff_var	stratification variable to be used when calculating deff.
domain_var	the variable used to identify the domain of interest.
minimum	minimum number of SSU to be selected from a PSU.
delta	average number of analysis units per sampling unit.
deff_sugg	suggested value of the deff.

Value

A list containing (i) the vector of allocated PSUs in the iterations and (ii) the vector of allocated SSUs in the iterations

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
# parameters
samp_frame <- pop
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")
deff_var <- "stratum"
domain_var <- "region"
minimum <- 50 # minimum number of SSUs to be interviewed in each selected PSU
# average dimension of the SSU in terms of elementary survey units
# delta =  nrow(pop) /length(unique(pop$id_hh))
delta =  1
deff_sugg <- 1.5
# prepare inputs
inp <- prepareInputToAllocation1(samp_frame,
```

```
                                id_PSU,
                                id_SSU,
                                strata_var,
                                target_vars,
                                deff_var,
                                domain_var,
                                minimum,
                                delta,
                                deff_sugg)

## End(Not run)
```

`prepareInputToAllocation2` *Input dataframes for R2BEAT two-stages sample design when both sampling frame and a previous round of the survey are available*

Description

In case of scenario 2 (at least one previous round of the survey available), prepares the following input dataframes for R2BEAT two-stages sample design starting from the sampling frame: 1. strata 2. deff 3. effst 4. rho 5. PSU_file 6. des_file

Usage

```
prepareInputToAllocation2 (
  samp_frame,
  RGdes,
  RGcal,
  id_PSU,
  id_SSU,
  strata_var,
  target_vars,
  deff_var,
  domain_var,
  delta,
  minimum)
```

Arguments

`samp_frame`	The dataframe containing sampling units in the reference population.
`RGdes`	The 'design' ReGenesees object.
`RGcal`	The 'calibration' ReGenesees object.
`id_PSU`	variables used as identifiers in sampling frame.
`id_SSU`	variables used as identifiers in sampling frame.
`strata_var`	stratification variable used in sampling frame.

target_vars	target variables.
deff_var	stratification variable to be used when calculating deff.
domain_var	the variable used to identify the domain of interest.
delta	average number of analysis units per sampling unit.
minimum	minimum number of SSU to be selected from a PSU.

Value

A list containing: (1) strata, (2) deff, (3) effst, (4) rho, (5) PSU_file, (6) des_file

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
samp <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/sample.rds?raw=true")
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
str(samp)
## Sample design description
library(ReGenesees)
samp$stratum_2 <- as.factor(samp$stratum_2)
sample.des <- e.svydesign(samp,
                          ids= ~ municipality + id_hh,
                          strata = ~ stratum_2,
                          weights = ~ weight,
                          self.rep.str = ~ SR,
                          check.data = TRUE)
## Find and collapse lonely strata
ls <- find.lon.strata(sample.des)
if (!is.null(ls)) sample.des <- collapse.strata(sample.des)
## Calibration with known totals
totals <- pop.template(sample.des,
             calmodel = ~ sex : cl_age,
             partition = ~ region)
totals <- fill.template(pop, totals, mem.frac = 10)
sample.cal <- e.calibrate(sample.des,
                          totals,
                          calmodel = ~ sex : cl_age,
                          partition = ~ region,
                          calfun = "logit",
                          bounds = c(0.3, 2.6),
                          aggregate.stage = 2,
                          force = FALSE)
samp_frame <- pop
RGdes <- sample.des
RGcal <- sample.cal
strata_var <- c("stratum")
```

```
target_vars <- c("income_hh",
                 "active",
                 "inactive",
                 "unemployed")
weight_var <- "weight"
deff_var <- "stratum"
id_PSU <- c("municipality")
id_SSU <- c("id_hh")
domain_var <- c("region")
delta <- 1
minimum <- 50
inp <- prepareInputToAllocation2 (
  samp_frame,
  RGdes,
  RGcal,
  id_PSU,
  id_SSU,
  strata_var,
  target_vars,
  deff_var,
  domain_var,
  delta,
  minimum)
head(inp$strata)
head(inp$deff)
head(inp$effst)
head(inp$rho)
head(inp$psu_file)
head(inp$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

```
## Not run:
# Load example data
data(beat.example)
PSU_strat
str(PSU_strat)

## End(Not run)
```

---

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

```
## Not run:
# Load example data
data(beat.example)
rho
str(rho)

## End(Not run)
```

`select_PSU` *Select sample of primary stage units (PSU)*

Description

Select sample of primary stage units (PSU) on the basis of the PSU allocated in the allocation step.

Usage

```
select_PSU(alloc, type="ALLOC", pps=TRUE, plot=TRUE)
```

Arguments

`alloc`	Output of the allocation step (beat.st)
`type`	Type of SSU allocation ("ALLOC" = optimal, "PROP" = proportional to population size, "EQUAL" = equal size in each stratum)
`pps`	Type of PSU selection in strata (pps = TRUE –> proportional to size, pps= FALSE–> simple random sampling)
`plot`	If TRUE, a plot of PSUs and SSUs ditribution is produced

Value

A list including: universe_PSU, sample_PSU

Author(s)

Alessio Guandalini

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.06),
                         CV3=c(0.03,0.06),
                         CV4=c(0.05,0.08)))
cv
samp_frame <- pop
samp_frame$one <- 1
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")
deff_var <- "stratum"
domain_var <- "region"
delta =  1       # households = survey units
minimum <- 50    # minimum number of SSUs to be interviewed in each selected PSU
deff_sugg <- 1.5 # suggestion for the deff value

inp <- prepareInputToAllocation1(samp_frame,
                                 id_PSU,
                                 id_SSU,
                                 strata_var,
                                 target_vars,
                                 deff_var,
                                 domain_var,
                                 minimum,
                                 delta,
                                 deff_sugg)
inp$desfile$MINIMUM <- 50
alloc <- beat.2st(stratif = inp$strata,
                  errors = cv,
                  des_file = inp$des_file,
                  psu_file = inp$psu_file,
                  rho = inp$rho,
                  deft_start = NULL,
                  effst = inp$effst,
                  minPSUstrat = 2,
                  minnumstrat = 50
)
```

```
sample_1st <- select_PSU(alloc, type="ALLOC", pps=TRUE, plot=TRUE)
sample_1st$PSU_stats

## End(Not run)
```

`select_PSU2` *Select sample of primary stage units (PSU)*

Description

Select sample of primary stage units (PSU) on the basis of the PSU allocated in the allocation step. This function differs from 'selectPSU' in that PSUs are not organized in sub-strata, but directly sampled with probability proportional to size in each sampling stratum. It also allows an implicit stratification by giving a set of ordering variables for each PSU.

Usage

```
select_PSU2(alloc,
type="ALLOC",
var_ord=NULL,
des_file=des_file,
psu_file = psu_file)
```

Arguments

`alloc`	Output of the allocation step (beat.st)
`type`	Type of SSU allocation ("ALLOC" = optimal, "PROP" = proportional to population size, "EQUAL" = equal size in each stratum)
`var_ord`	dataframe containing for all PSUs a set of variables to be used to order the PSUs
`des_file`	dataframe containing information on sampling strata
`psu_file`	dataframe containing the list of Primary Sampling Units

Value

list containing: (i) information on sampled PSUs (ii) list of selected PSUs

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.06),
                         CV3=c(0.03,0.06),
                         CV4=c(0.05,0.08)))
cv
samp_frame <- pop
samp_frame$one <- 1
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")
deff_var <- "stratum"
domain_var <- "region"
delta =  1       # households = survey units
minimum <- 50    # minimum number of SSUs to be interviewed in each selected PSU
deff_sugg <- 1.5 # suggestion for the deff value

inp <- prepareInputToAllocation1(samp_frame,
                                 id_PSU,
                                 id_SSU,
                                 strata_var,
                                 target_vars,
                                 deff_var,
                                 domain_var,
                                 minimum,
                                 delta,
                                 deff_sugg)
inp$desfile$MINIMUM <- 50
alloc <- beat.2st(stratif = inp$strata,
                  errors = cv,
                  des_file = inp$des_file,
                  psu_file = inp$psu_file,
                  rho = inp$rho,
                  deft_start = NULL,
                  effst = inp$effst,
                  minPSUstrat = 2,
                  minnumstrat = 50
)
sample_1st <- select_PSU2(alloc, type="ALLOC", var_ord=NULL, des_file=des_file)
head(sample_1st)

## End(Not run)
```

`select_SSU` *Select sample of secondary stage units (SSU)*

Description

Select sample of secondary stage units (SSU) from the population frame on the basis of the SSU allocated to each selected PSU

Usage

```
select_SSU(df,PSU_code,SSU_code,PSU_sampled)
```

Arguments

df	Dataframe containing sampling units (SSUs)
PSU_code	Identifier of each PSU in dataframe containing sampling units
SSU_code	Identifier of each SSU in dataframe containing sampling units
PSU_sampled	Dataframe containing selected PSUs

Value

A dataframe containing the units selected in the sample

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.06),
                         CV3=c(0.03,0.06),
                         CV4=c(0.05,0.08)))
cv
samp_frame <- pop
samp_frame$one <- 1
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")
deff_var <- "stratum"
domain_var <- "region"
delta =  1       # households = survey units
minimum <- 50    # minimum number of SSUs to be interviewed in each selected PSU
deff_sugg <- 1.5 # suggestion for the deff value

inp <- prepareInputToAllocation1(samp_frame,
                                 id_PSU,
                                 id_SSU,
```

```
                                    strata_var,
                                    target_vars,
                                    deff_var,
                                    domain_var,
                                    minimum,
                                    delta,
                                    deff_sugg)
inp$desfile$MINIMUM <- 50
alloc <- beat.2st(stratif = inp$strata,
                  errors = cv,
                  des_file = inp$des_file,
                  psu_file = inp$psu_file,
                  rho = inp$rho,
                  deft_start = NULL,
                  effst = inp$effst,
                  minPSUstrat = 2,
                  minnumstrat = 50
)
sample_1st <- select_PSU(alloc, type="ALLOC", pps=TRUE, plot=TRUE)
samp <- select_SSU(df=pop,
                  PSU_code="municipality",
                  SSU_code="id_ind",
                  PSU_sampled=sample_1st$sample_PSU)

## End(Not run)
```

`sensitivity_min_SSU` *Sensitivity analysis for choosing minimum number of SSUs per PSU (sampling frame available)*

Description

This function allows to analyse the different results in terms of first stage size (number of PSUs) and second stage size (number of SSUs), when varying the values of the minimum number of SSU per single PSU) The name of the parameter has to be given, together with the minimum and maximum value. On the basis of these minimum and maximum values, 10 different values will be used for carrying out the allocation. The output will be a graphical one. To be used only in the scenario when no previous rounds of the survey are available, and a frame complete with values of target variables is available.

Usage

```
sensitivity_min_SSU (samp_frame,
                     errors,
                     id_PSU,
                     id_SSU,
                     strata_var,
                     target_vars,
                     deff_var,
```

```
                    domain_var,
                    delta,
                    deff_sugg,
                    min,
                    max,
                    plot)
```

Arguments

samp_frame	The dataframe containing sampling units in the reference population.
errors	Precision constraints.
id_PSU	variables used as identifiers in sampling frame.
id_SSU	variables used as identifiers in sampling frame.
strata_var	stratification variable used in sampling frame.
target_vars	target variables.
deff_var	stratification variable to be used when calculating deff.
domain_var	the variable used to identify the domain of interest.
delta	average number of analysis units per sampling unit.
deff_sugg	suggestion for deff value.
min	minimum value of the parameter.
max	maximum value of the parameter.
plot	plot (TRUE/FALSE) the final result.

Value

A list containing the (i) vector of allocated PSUs in the iterations and (ii) the vector of allocated SSUs in the iterations

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.03,0.04),
                         CV2=c(0.06,0.08),
                         CV3=c(0.06,0.08),
                         CV4=c(0.06,0.08)))
cv
# parameters
samp_frame <- pop
errors <- cv
```

```
id_PSU <- "municipality"
id_SSU <- "id_ind"
strata_var <- "stratum"
target_vars <- c("income_hh","active","inactive","unemployed")   # more than one
deff_var <- "stratum"
domain_var <- "region"
# average dimension of the SSU in terms of elementary survey units
#delta =  nrow(pop) /length(unique(pop$id_hh))
delta =  1  # average dimension of the SSU in terms of elementary survey units
deff_sugg <- 1.5  # deff (suggested)
sensitivity_min_SSU(
            samp_frame,
            errors,
            id_PSU,
            id_SSU,
            strata_var,
            target_vars,
            deff_var,
            domain_var,
            delta,
            deff_sugg,
            min=1,
            max=2)

## End(Not run)
```

`sensitivity_min_SSU2` *Sensitivity analysis for choosing minimum number of SSUs per PSU (no sampling frame available)*

Description

This function is similar to the function sensitivity_min_SSU, as it allows to analyse the different results in terms of first stage size (number of PSUs) and second stage size (number of SSUs), when varying the values of the minimum number of SSU per single PSU) The name of the parameter has to be given, together with the minimum and maximum value. Differently from sensitivity_min_SSU, it requires in input all the outputs already prepared by prepareInputToAllocation1 or prepareInputToAllocation2.

Usage

```
sensitivity_min_SSU2 (strata,
                      des_file,
                      psu_file,
                      rho,
                      effst,
                      errors,
                      min,
                      max,
                      plot)
```

Arguments

strata	Data frame of survey strata, for more details see, e.g.,strata.
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.
effst	Data frame of survey strata, for taking into account the estimator effect on each variable, for more details see, e.g.,effst.
errors	Data frame of coefficients of variation for each domain, for more details see, e.g.,errors.
min	starting value for the minimum value of SSUs per PSU.
max	ending value for the minimum value of SSUs per PSU.
plot	plot (TRUE/FALSE) the final result.

Value

A list containing the (i) vector of allocated PSUs in the iterations and (ii) the vector of allocated SSUs in the iterations

Author(s)

Giulio Barcaroli

Examples

```
## Not run:
library(readr)
samp <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/sample.rds?raw=true")
pop <- read_rds("https://github.com/barcaroli/R2BEAT_workflows/blob/master/pop.RDS?raw=true")
library(R2BEAT)
str(samp)
## Sample design description
library(ReGenesees)
samp$stratum_2 <- as.factor(samp$stratum_2)
sample.des <- e.svydesign(samp,
                          ids= ~ municipality + id_hh,
                          strata = ~ stratum_2,
                          weights = ~ weight,
                          self.rep.str = ~ SR,
                          check.data = TRUE)
## Find and collapse lonely strata
ls <- find.lon.strata(sample.des)
if (!is.null(ls)) sample.des <- collapse.strata(sample.des)
## Calibration with known totals
totals <- pop.template(sample.des,
             calmodel = ~ sex : cl_age,
             partition = ~ region)
```

```
totals <- fill.template(pop, totals, mem.frac = 10)
sample.cal <- e.calibrate(sample.des,
                          totals,
                          calmodel = ~ sex : cl_age,
                          partition = ~ region,
                          calfun = "logit",
                          bounds = c(0.3, 2.6),
                          aggregate.stage = 2,
                          force = FALSE)
samp_frame <- pop
RGdes <- sample.des
RGcal <- sample.cal
strata_var <- c("stratum")
target_vars <- c("income_hh",
                 "active",
                 "inactive",
                 "unemployed")
weight_var <- "weight"
deff_var <- "stratum"
id_PSU <- c("municipality")
id_SSU <- c("id_hh")
domain_var <- c("region")
delta <- 1
minimum <- 50
inp <- prepareInputToAllocation2 (
  samp_frame,
  RGdes,
  RGcal,
  id_PSU,
  id_SSU,
  strata_var,
  target_vars,
  deff_var,
  domain_var,
  delta,
  minimum)
cv <- as.data.frame(list(DOM=c("DOM1","DOM2"),
                         CV1=c(0.02,0.03),
                         CV2=c(0.03,0.06),
                         CV3=c(0.03,0.06),
                         CV4=c(0.05,0.08)))
sens <- sensitivity_min_SSU2 (strata = inp$strata,
                         des_file = inp$des_file,
                         psu_file = inp$psu_file,
                         rho = inp$rho,
                         effst = inp$effst,
                         errors = cv,
                         min = 15,
                         max = 25,
                         plot = TRUE)

## End(Not run)
```

sens_names *Function for better presentation of sensitivity information*

Description

This function allows to obtain a better presentation of sensitivity information, i.e. with the names of the variables and of the domains

Usage

```
sens_names(s,target_vars,strata)
```

Arguments

s	The sensitivity output of the allocation step.
target_vars	target variables.
strata	strata dataframe used in the allocation step..

Value

A dataframe containing Sensitivity information

Author(s)

Giulio Barcaroli

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

```
## Not run:
# Load example data
data(beat.example)
strata
str(strata)

## End(Not run)
```

Index