Two-Stage Sampling Design and Sample Selection with the R Package R2BEAT

by Giulio Barcaroli, Andrea Fasulo, Alessio Guandalini, and Marco D. Terribili

Abstract R2BEAT ("R 'to' Bethel Extended Allocation for Two-stage sampling") is an R package for the optimal allocation of a sample. Its peculiarity lies in properly addressing allocation problems for two-stage and complex sampling designs with multi-domain and multi-purpose aims. This is common in many official and non-official statistical surveys, therefore R2BEAT could become an essential tool for planning a sample survey. The functions implemented in R2BEAT allow the use of different workflows, depending on the available information on one or more interest variables. The package covers all the phases, from the optimization of the sample to the selection of the Primary and Secondary Stage Units. Furthermore, it provides several outputs for evaluating the allocation results.

1 Introduction

Sample surveys carried out by National Statistical Institutes (NSIs) and by other institutions have multi-domain and multi-purpose objectives, so they have to provide accurate estimates for different parameters and different domains (i.e. geographical areas such as national, regional, and more).

Surveys have budgetary and logistical constraints, so they must be carefully planned to provide high-quality estimates for parameters of interest. In this context, several decisions need to be made, such as sample size, stratification, allocation of sampling units among the strata and multiple-stage.

These decisions may not be trivial and can strongly affect all the following steps of the survey and, moreover, the quality of the results. This justifies the care and attention typically given in the literature. A seminal work in this perspective is that of Kish (1965) and, later, different approaches have been proposed for defining optimal sampling strategies, i.e. maximizing data quality within budgetary constraints (see among the others Cochran, 1977).

The proposed package, R2BEAT (standing for R "to" Bethel Extended Allocation for Two-stage), fits into this context and fills a gap within the range of statistical software concerning sample size allocation, thereby introducing an improvement into the R community.

There are several R packages for allocating a stratified sample, such as surveyplanning (Breidaks et al., 2020), PracTools (Valliant et al., 2020), samplesize4surveys (Rojas, 2020), optimStrat (Bueno, 2020) and SamplingStrata (Barcaroli, 2014). But R2BEAT, extending the methodology implemented in the Italian National Institute of Statistic's open-source software called Mauss-R ("Multivariate Allocation of Units in Sampling Surveys") (Barcaroli et al., 2020), enables us also to compute the optimal allocation among strata for two-stage and complex sampling design considering both multivariate and multi-domain cases.

In the following section the methodological aspects, underlying the package and its functions, will be presented in detail: the optimal allocation of the sample and its selection will be illustrated. In the third section will be shown how to prepare, organize and check the input data needed by the package for allocating the whole sample size among strata and finally select the units. A case study on a synthetic dataset will be used as an example to test the package functions, and also for a comparison to the other two packages handling two-stage sample design (PracTools and samplesize4surveys). Finally, the concluding remarks will point out that R2BEAT can allocate the sample more efficiently than the other available multistage allocation software, while also being strongly flexible, generalizable, and integrated with software that manages all the different phases of the statistical data production process.

2 Methodological aspects

2.1 The optimal allocation

Let us consider a population U of size N ($k = 1, \ldots, N$) partitioned in H subgroups, U_h ($h = 1, \ldots, H$), called strata. Hence, each stratum contains N_h elements, where N_h is assumed to be known such that $\sum_{h=1}^{H} N_h = N$.

The strata can be defined in different ways on the basis of one or more qualitative variables known for all the units in the population.

Then, we assume, at least for the moment, to be interested in investigating the mean of just one y variable in the population U,

$$\mu_y = \frac{\sum_{k \in U} y_k}{N} \tag{1}$$

where y_k is the value of the y variable observed on the k-th unit in the population U. The y variable could be a quantitative variable or dichotomous, that is $y \in \{0, 1\}$. Please note that, even when y is a dichotomous variable, expression (1) holds and μ_y is equal to the proportion of units in the population for which $y = 1$.

Furthermore, assume we want to estimate μ_y through a probabilistic sample s of size n with the estimator

$$\hat{\bar{Y}} = \frac{\hat{Y}_{HT}}{N} = \frac{\sum_{k \in s} y_k \, d_k}{N} \tag{2}$$

where $\hat{Y}_{HT}$ is the Horvitz-Thompson estimator for the total (Horvitz and Thompson, 1952) in which d_k is the design weight usually equal to the inverse of the first order inclusion probability.

The sample size of a survey, n, is usually exogenous information, dictated by budget and, sometimes, by logistic constraints associated with the unit k in the sample. Then, in practice, the problem comes down to the allocation of the n units in the H strata, such that $\sum_{h=1}^{H} n_h = n$.

Therefore, let us define

$$\mu_{hy} = \frac{\sum_{U} y_k \, \mathbf{1}_h}{N_h} \tag{3}$$

the mean of the y in each stratum where $\mathbf{1}_h$ is the membership indicator for the unit k in the stratum h.

In the same way, expression (2) can be easily adapted for estimating μ_{hy}, that is

$$\hat{\bar{Y}}_h = \frac{\hat{Y}_{HT,h}}{N_h} = \frac{\sum_{k \in s_h} y_k \, d_k}{N_h}, \tag{4}$$

where s_h is the sample in the stratum h. The sampling variance estimator of $\hat{\bar{Y}}_h$ is given by

$$\widehat{\text{var}}\left(\hat{\bar{Y}}_h\right) = \frac{1}{n_h - 1}\left(\frac{1}{n_h} - \frac{1}{N_h}\right)\sum_{k \in h}(y_{hk} - \bar{y}_h)^2, \tag{5}$$

where $\bar{y}_h$ is the sample mean of the variable y in the stratum h.

The mean of y in (1) can be written also as $\mu_y = \sum_{h=1}^{H} (N_h/N)\, \mu_{hy}$ and, consequently, $\hat{\bar{Y}}$ in (2) as $\hat{\bar{Y}} = \sum_{h=1}^{H} (N_h/N)\, \hat{\bar{Y}}_h$. Therefore, the sampling variance estimator for $\hat{\bar{Y}}$ is $\widehat{\text{var}}\left(\hat{\bar{Y}}\right) = \sum_{h=1}^{H} (N_h/N)^2 \, \widehat{\text{var}}\left(\hat{\bar{Y}}_h\right)$.

When there is no information on y, the sample size to be allocated to each stratum, n_h, can be assigned by performing uniform or proportional allocation.

Uniform allocation assigns an equal number of sampling units to each stratum, that is $n_h^{UNIF} = n/H$.

More often, we want the sample size assigned to strata in the sample to be proportional to the sizes of the strata in the population, that is $n_h^{PROP} = n \ (N_h/N)$ where N_h/N is the weight of the stratum in the population with $\sum_{h=1}^{H} N_h/N = 1$. If the size is the same for all strata ($N_1 = \cdots = N_h = \cdots = N_H = N/H$), n_h^{PROP} comes down to n_h^{UNIF}.

When there is information in the population strata on y and in particular on its variance, S_{yh}^2, a more favorable allocation can be performed. Alternatively, it is possible to consider also a proxy variable highly correlated with y. In this case, Tschprow (1923) demonstrated that the optimal allocation can be obtained by

$$n_h^{OPT} = n \frac{\frac{N_h}{N}\sqrt{S_{yh}^2}}{\sum_{h=1}^{H} \frac{N_h}{N}\sqrt{S_{yh}^2}}$$

However, this result, also published by Neyman (1934), is more often referred to as Neyman's allocation. The rationale behind the optimal allocation is that strata with more weight and in which y has much more variability need many more observations to reach better estimates. If the variance is the same in all the strata ($S_1^2 = \cdots = S_h^2 = \cdots = S_H^2$), n_h^{OPT} comes down to n_h^{PROP}.

The computation of the population variance is a crucial point in the optimal allocation. A dis-

tinction between the types of variables and the sources from which they can be obtained is needed. When y is a dichotomous variable available from a population register, its population variance can be computed as

$$S_{yh}^2 = p_h \times (1 - p_h) \tag{6}$$

where p_h is the proportion of units with $y = 1$ in the population strata. In the case of a quantitative variable, S_{yh}^2 is equal to $S_{yh}^2 = \left[\sum_{k \in U_h} \left(y_k - \mu_{yh}\right)^2\right] / N_h$.

When there is no population register, information on the variability can be obtained from a sample survey or a pilot survey which has previously been carried out. Let us assume to have collected the y variable, or at least its proxy variable, on a sample s^*. Then, (6) can be computed just by replacing p_h with

$$\hat{p}_h = \frac{\sum_{k \in s_h^*} y_k \, w_k}{\sum_{k \in s_h^*} w_k}, \tag{7}$$

that is the related estimate for each stratum obtained from the sample s^*. In (7) w_k is the sampling weight associated with the unit k in the sample s^*.

Instead, when y is a quantitative variable, $S_{yh}^2 = \hat{M}_h^2 - \hat{\bar{Y}}_h^2$ where $\hat{M}_h^2 = \left(\sum_{k \in s_h^*} y_k^2 \, w_k\right) / N$ and $\hat{\bar{Y}}_i = \left(\sum_{k \in s_h^*} y_k \, w_k\right) / N$ are the quadratic mean and the arithmetic mean estimated on the sample s^* in the h-th stratum, respectively.

The optimal allocation for just one y variable is of little practical use unless the various variables under study are highly correlated. This is because an allocation that is optimal for one characteristic is generally far from being optimal for others.

Therefore, several works have been devoted to solving the problem when more than one variable of interest has to be measured on each sampled unit. All the contributions can be classified into two main approaches: the “average variance" and convex programming.

The methods under the “average variance" approach consist of defining a weight for each variable to consider, computing a weighted average of the stratum variance and finding the optimal allocation using the “average variance" which results. They are computationally simple, intuitive and can be solved under fixed cost assumption. However, the choice of the weights is completely arbitrary and the optimal properties are not clear (see, e.g., Kish, 1976, for more details).

Instead, the other approach includes methods that use convex programming to find the minimum cost allocation, satisfying the fixed constraints regarding the variances of all the sampling variables. The obtained allocation is actually optimal, but sometimes it can exceed the budgetary constraints.

The most important method in the convex programming approach is the Bethel algorithm (Bethel, 1989) which extends the Neyman allocation to the multivariate case, providing the optimal allocation in the strata, in terms of surveying cost, according to a set of variables observed in a multidomain context.

In particular, when we are interested in investigating the mean of more than one y variable (quantitative or dichotomous), namely $y_1, \ldots, y_i, \ldots, y_I$, the optimal allocation problem reduces, in practice, to a minimum optimization problem of a convex function under a set of linear constraints

$$\begin{cases} C = \min \\ \widehat{CV}\left(\hat{\bar{Y}}_{i,h}\right) \leq \delta\left(\hat{\bar{Y}}_{i,h}\right) & \begin{matrix} i = 1, \ldots, I \\ h = 1, \ldots, H \end{matrix} \end{cases} \tag{8}$$

where C is the global cost of the survey to be minimized and $\hat{CV}\left(\hat{\bar{Y}}_{i,h}\right)$ is the estimate of the relative error. The estimate of the relative error,

$$\widehat{CV}\left(\hat{\bar{Y}}_{i,h}\right) = \frac{\sqrt{\widehat{\text{var}}\left(\hat{\bar{Y}}_{i,h}\right)}}{\hat{\bar{Y}}_{i,h}}, \tag{9}$$

is the ratio between the estimate of the sampling variance for the mean estimator of y_i variable ($i = 1, \ldots, I$) in the stratum h given by expression (5) and the related estimate. In this case, $\widehat{CV}\left(\hat{\bar{Y}}_{i,h}\right)$ is called expected error and it must be less than or equal to the precision constraints defined by the user or by regulation, $\delta\left(\hat{\bar{Y}}_{i,h}\right)$.

Bethel (1989) demonstrates that the solution to this optimization problem exists and can be obtained

through an algorithm that applies the Lagrange multipliers method. The solution is a real number, so it must be rounded to provide an integer stratum sample size. The rounding clearly causes some deviations from the solution that, however, do not affect its optimality (Cochran, 1977).

This framework also works in the case of the multi-domain problem. Usually, estimates of a survey are disseminated for the whole population and sub-domains, for instance for geographical areas. Then, it is useful to define the optimal allocation also taking into account these outcomes of the survey.

Sub-domain estimation is actually a long-established theory (Särndal et al., 2003). Expression (1) can be easily adapted just by introducing the sub-domain membership indicator variable, $\mathbf{1}_{k,d}$, which equals 1 for each the unit k in the domain d and 0 otherwise, that is $\mu_y^d = \left(\sum_{U_d} y_k \, \mathbf{1}_{k,d}\right) / N_d$ where N_d is the population size in the domain d ($d = 1, \ldots, D$). It is important to point out, that domains must be an aggregation of strata and thus should not split strata. Then, it is sufficient to consider the domain estimates in the minimum optimization problem in (8) and use Bethel's algorithm for deriving the multivariate allocation in the multi-domain case.

However, in official statistics, especially for household surveys, two-stage sampling designs are usually adopted. Two-stage sampling is based on a double sampling procedure: one on the primary stage units (PSUs) and another on the second stage units (SSUs). For instance, in the household survey, the PSUs are the municipalities, which are selected first. Then, in each selected municipality, a sample of households - the *SSU* - can be selected.

Two-stage sampling permits more complex sampling strategies and, moreover, it helps in the organization and cost reduction of data collection, because it reduces the interviewer's travels. However, this economic saving is counterbalanced with a loss of efficiency of the estimates. In fact, each additional stage of selection usually entails an increase of the sampling variance of the mean estimator. This increase can be assessed by the design effect ($deff$) that measures how much the sampling variance of $\hat{\bar{Y}}_i$, under the adopted sampling design (des), is inflated with respect to a simple random sample (srs), with the same sample size. An estimate of the design effect can be given by the expression:

$$deff\left(\hat{\bar{Y}}_i\right) \quad = \quad \widehat{\text{var}}\left(\hat{\bar{Y}}_i\right)_{des} / \widehat{\text{var}}\left(\hat{\bar{Y}}_i\right)_{srs}$$

A rough approximation of the $deff$ can be obtained when the clusters have the same sample size and the same inclusion probability (Cicchitelli et al., 1992),

$$deff\left(\hat{\bar{Y}}_i\right) \quad = \quad 1 + \rho_i \, (b - 1) \tag{10}$$

where b is the average cluster (i.e. PSU) size in terms of the final sampling units and ρ_i is the intra-class correlation within the cluster (PSU) for the variable y_i $(i = 1, \ldots, I)$.

The intra-class correlation provides a measure of data clustering in PSUs and SSUs. In general, if ρ_i is close to 1, the clustering is high and it is convenient to collect only a few units in the cluster. Instead, if ρ_i is close to 0, the collection of units from the same cluster does not affect the efficiency of the estimates.

Also for computing ρ_i, we can distinguish whether a population register in which the y_i variables ($i = 1, \ldots, I$), or at least their proxies, are available or not. In the former case, a good approximation, given in Cicchitelli et al. (1992), is

$$\rho_i = 1 - \left(D_{w_i} / D_{y_i}\right) \tag{11}$$

where $D_{w_i} = \sum_{h=1}^{H} \sum_{k=1}^{N_h} \left(y_{i,k} - \mu_{y_{i,h}}\right)^2$ is the deviance within clusters and $D_{y_i} = \sum_{k \in U} \left(y_{i,k} - \mu_{y_i}\right)^2$ is the global deviance of the y_i variable. Remember that $D_{y_i} = D_{w_i} + D_{b_i}$, where $D_{b_i} = \sum_{h=1}^{H} N_h \left(\mu_{yi,h} - \mu\right)^2$, is the deviance between clusters. Therefore, $0 \leq \rho_i \leq 1$.

Instead, ρ_i can be estimated from a sample with the expression (10)

$$\hat{\rho}_i = \left(deff_i - 1\right) \, (b - 1) \, . \tag{12}$$

Here we consider, directly, a more general expression for the estimate of the $deff$ in terms of the intra-class correlation coefficient. This expression refers to a typical situation in household surveys where PSUs are assigned to Self-Representing (SR) strata, that is they are included for sure in the sample, or to Not-Self-Representing (NSR) strata, where they are selected by chance. In practice, this assignment is usually performed by comparing the measure of the size of PSUs to the threshold (see, e.g., Hansen et al., 1953):

$$\lambda = (\bar{m} \, \Delta) / f \tag{13}$$

where $\bar{m}$ is the minimum number of SSUs to be interviewed in each selected PSU, $f = n/N$ is the sampling fraction and Δ is the average dimension of the SSU in terms of elementary survey units.

Then, Δ must be set to 1 if, for the survey, the selection units are the same as the elementary units (that is, household-household or individuals-individuals), whereas it must be set equal to the average dimension of the households if the elementary units are individuals, while the selection units are the households. PSUs with a measure of size exceeding the threshold are identified as SR, while the remaining PSUs are identified as NSR.

Then, the extended expression of $deff$ (see, among others, Rojas, 2016) is

$$deff\left(\hat{\bar{Y}}_i\right) = \frac{n}{N^2}\left\{\frac{N_{SR}^2}{n_{SR}}\left[1+\left(\rho_{i,SR}\left(b_{SR}-1\right)\right)\right]+\frac{N_{NSR}^2}{n_{NSR}}\left[1+\left(\rho_{i,NSR}\left(b_{NSR}-1\right)\right)\right]\right\} \tag{14}$$

where, for SR and NSR strata,

- N_{SR} and N_{NSR} are the population sizes;
- n_{SR} and n_{NSR} are the sample sizes;
- $\rho_{i,SR}$ and $\rho_{i,NSR}$ are the intra-class correlation coefficients for the variable i ($i = 1, \ldots, I$);
- b_{SR} and b_{NSR} are the average PSU size in terms of the final sampling units.

Of course, if there are no SR strata, this expression simplifies to Equation (10). The design effect is equal to 1 under the srs design and increases for each additional stage of selection, due to the intra-class correlation coefficient which is, usually, positive.

The intra-class correlation coefficient for NSR can be computed with expression (11) or (12) whether population register data are available or not. It is not necessary to compute the intra-class correlation coefficient for SR strata because just one PSU is selected and the intra-class correlation is 1 by definition.

Therefore, under a two-stage sample design for determining the optimal allocation, the number of PSUs and SSUs must be determined.

A solution which makes iterative use of the Bethel algorithm has been proposed by Falorsi et al. (1998). In fact, at the first iteration, the Bethel algorithm is applied. The optimal allocation for a stratified simple sampling design is obtained. Then, this allocation is used to update the threshold in (13) and the design effect in (14). A new design effect is computed and used in turn to inflate the S_h^2 (or equivalently $\hat{S}_h^2$). It is used as input in the next iteration in which the Bethel algorithm is used again. The obtained allocation is used again to update the threshold and the design effect, and a new allocation is found. The process is iterated until the difference between two consecutive iterations is lower than a predefined threshold.

However, as pointed out by Waters and Chester (1987), different combinations yield the same variance and can satisfy the precision constraints, $\delta\left(\hat{\bar{Y}}_{i,h}\right)$. The optimal solution strongly depends on the budgetary constraints that limit the SSUs and the data collection organization that influences the maximum number of PSUs that can be managed.

All this discussion holds when you want to use the HT estimator. But, currently, the most applied estimator for the NSIs survey is the calibrated estimator (Deville and Särndal, 1992). The calibrated estimator, through the use of auxiliary variables, usually provides better estimates than HT. The use of a different estimator from the HT can be considered since the allocation phase, by accounting for the estimator effect and following the procedure explained above

An estimate of the estimator effect ($effst$) is given by

$$effst(\hat{\bar{Y}}_i) = \frac{\text{var}\left(\hat{\bar{Y}}_i\right)}{\text{var}\left(\hat{\bar{Y}}_{i,HT}\right)}. \tag{15}$$

It measures how much the sampling variance of the applied estimator under the adopted design is inflated or deflated with respect to the sampling variance of the HT estimator, on the same sample design.

2.2 Sample selection

Once the optimal allocation is defined, the selection of sampling units must be performed.

In the case of a stratified two-stage sampling design two sampling selections need to be done: one for PSUs and one for SSUs.

In each stratum, the PSUs are split into SR and NSR according to a size threshold (13). PSUs with a measure of size exceeding the threshold are identified as SR, included for sure in the sample and each

Algorithm 1: R2BEAT optimal allocation of PSUs and SSUs in sampling strata

Input :

a. precision constraints in terms of CV;

b. information on sampling strata (mean and stdev of target variables, N, ...);

c. information on previous design: $deff$, $effst$, ρ;

d. information on PSUs in sampling strata (measure of size);

e. minimum number of SSUs per PSU;

Output:

a. for each stratum: number of PSUs and SSUs to be selected;

b. expected CVs for target estimates;

c. sensitivity of the solution;

REM First iteration;

1. input deff is used to inflate standard deviations of target variables in sampling strata;
2. optimal allocation of SSUs in sampling strata is obtained by applying the Bethel algorithm as if it were a one-stage sampling design;
3. the number of PSUs is determined on the basis of the minimum number of SSUs per PSU;
4. the threshold for determination of self-representing PSUs is calculated;
5. new deff is calculated and used to update the standard deviations of target variables in sampling strata;

REM Next iterations;
while *not convergence* **do**

1. optimal allocation of SSUs in sampling strata is obtained by applying the Bethel algorithm;
2. the number of PSUs is determined on the basis of the minimum number of SSUs per PSU;
3. the threshold for determination of self-representing PSUs is calculated;
4. new deff is calculated an used to update standard deviations of target variables in sampling strata;
5. the iteration stops if
   a. the difference between the sample sizes of two iterations is lower than 5 (default value) *or*
   b. the maximum of defts (square root of deffs) largest differences is lower than 0.06 (default value) *or*
   c. the number of iterations is higher than 20 (default value);

end

of them constitutes an independent sub-stratum. Therefore, the probability that they are included in the sample (inclusion probability, π_I) is always equal to 1.

It is possible that that it can happen that no PSU has a measure of size higher than the threshold: this can happen for example when we consider as PSUs the census enumeration areas, whose distribution of the measure of size is about uniform; on the contrary, it is unlikely to happen when we consider municipalities.

The remaining PSUs, NSR-PSUs, are ordered by their measure of the size and divided into finer strata (*sub-strata*) whose sizes are approximately equal to the threshold multiplied by the number of PSUs to be selected in each stratum. In this way, sub-strata are composed of PSUs having size as homogeneous as possible.

The PSUs in each stratum can be selected in different ways. However, the selection of a fixed number of PSUs per stratum is usually carried out with Sampford's method (unequal probabilities, without replacement, fixed sample size).

Finally, the SSUs must be drawn in the selected PSU. Also in this case the SSU can be selected in different ways. In most cases, they are selected through a systematic sampling design that shares several properties with the *srs*.

Then, the design weight for the unit k in the h-th strata in the ℓ-th PSU is equal to the inverse of the product of the first stage and the second stage inclusion probabilities, $d_k = \frac{1}{\pi_l}\frac{1}{\pi_{ll}}$. The design weights sum up to the population size, $\sum_{k\in s} d_k = N$, and are almost constant in each stratum, which means that the sample is self-weighting.

Algorithm 2: R2BEAT selection of PSUs

Input :

a. The output of the allocation step (function `beat.2st`) (universe of PSUs, measure of PSUs, number of PSUs and SSUs to be selected in each stratum, threshold);

Output:

a. universe of PSUs with stratum, sub-stratum, PSU first order inclusion probability, PSU weight, flag sample, and number of SSUs to be selected in each PSU;

b. sample of PSUs (flag sample=1) with stratum, sub-stratum, PSU first order inclusion probability, PSU weight, number of SSUs to be selected in each PSU;

c. statistics related to the sample of PSUs at stratum level;

REM creation of *sub-strata* and selection of PSUs;

1. in each stratum, PSUs are sorted in descending order according to their measure of size;
2. the measure of size of PSUs are compared with the threshold;
3. PSUs with a measure of size exceeding the threshold are identified as SR, included for sure in the sample and constitute an independent sub-stratum;
4. the remaining PSUs, NSR-PSUs, are ordered in decreasing way by their measure of size and aggregated into finer strata (*sub-strata*);
5. *sub-strata* are created adding PSUs (still in descending order of measure of size) for which the sum of the measure of size of the *sub-strata* is approximately equal to the threshold multiplied by the number of PSUs to be selected in each stratum;
6. in each *sub-stratum* a fixed number of PSUs per stratum are usually selected with Sampford's method (unequal probabilities, without replacement, fixed sample size);

3 Structure of the package

The R2BEAT package provides functions for drawing complex sample designs using an optimal allocation also performing the selection of the PSUs and SSUs. To install the latest release version of R2BEAT from CRAN, type **install.packages("R2BEAT")** within R. The current development version can be downloaded and installed from GitHub by executing **devtools::install_github("barcaroli/R2BEAT")**.

The workflow to draw and select a complex sample using R2BEAT is: (1) prepare the input data, (2) check the input data, (3) define the design and obtain the allocation, and (4) select the final sample units.

3.1 Prepare the input data

As will be illustrated in detail in the next sub-sections the R2BEAT package provides functions to define one-stage stratified sample design (**beat.1st**) and two-stage stratified sample design (**beat.2st**). The preparation of the input dataset changes whether the former or the latter sample design will be adopted.

In the case of a multivariate optimal allocation for different domains in a stratified one-stage sample design, the function **beat.1st** can be used. This function requires two inputs, a data frame containing survey strata information (**stratif**) and a data frame of expected CV for each domain and each variable (**errors**). No functions to prepare these inputs are provided by the package but is possible to follow the example dataset **stratif** and **error** to properly create the input datasets for the function **beat.1st**.

In the case of a two-stage design, two functions are provided by the package to help in the creation of the input data for the function **beat.2st**. There are two functions because two different scenarios are possible, depending on the initial information available:

1. Only the sampling frame is available, no previous rounds of the survey have been carried out. In this scenario, a strict condition on the information content of the sampling frame must hold: values of the sample target surveys (or of their proxy correlated variables) are available for each unit in the frame. This can be accomplished by considering the previous census, or by using administrative registers. In this scenario, the function **prepareInputToAllocation1** can be used to create the input dataframes **stratif**, **rho**, **deft**, **effst**, **des_file** and **psu_file**.

2. Together with a sampling frame containing the units of the population of reference, also a previous round of the sampling survey to be planned is available. The **prepareInputToAllocation2** produces the same outputs of **prepareInputToAllocation1**, but it requires the design and/or calibrated objects of the previous sample survey, obtained using the ReGenesees package (Zardetto, 2015).

The function **sensitivity_min_SSU** allows analyzing the different results in terms of first stage size (number of PSUs) and second stage size (number of SSUs), obtained when varying the values of the minimum number of SSUs to be selected in each PSU.

When using a previous survey, the 'strata' dataframe is automatically obtained by estimating all the variables in it, including the 'N', that is the population in strata. Being an estimate, these values can differ from the ones obtained by the 'des' dataframe (obtained by aggregating PSUs values). It is up to the user to decide if PSUs derived N values are more reliable than those obtained by the survey, and, in case, to assign those to the 'strata' dataframe. To check the coherence between the estimated population in the strata (**stratif**) and the population calculated by the PSUs dataset (**des_file**), the function **check_input** is provided to the users. This function compares the strata sizes giving information about the differences and replacing the estimated stratum size with the stratum population calculated by the PSUs dataset.

3.2 Defining the design and determining the allocation

The package allows performing the optimal allocation for both one-stage and two-stage stratified sampling.

The first one is implemented within the function **beat.1st** and computes a multivariate optimal allocation for different domains in a one-stage stratified sample design. As described in the previous section, in a one-stage stratified sample design there are only two inputs to be provided to **beat.1st**: the dataframes **stratif** and **errors**. Besides these two mandatory inputs, it is also possible to indicate the minimum number of sampling units to be selected in each stratum, by default set equal to 2.

The function **beat.2st** performs the same multivariate optimal allocation for different domains considering stratified two-stage design. Together with the input data **stratif** and **errors** other mandatory input are:

- **des_file**: dataframe containing a row per each stratum, with information on total population, the values of the **delta** parameter (equal to the mean number of final SSUs contained in clusters to be selected, for instance, the mean number of individuals in a household), and the minimum number of SSUs to be selected in each PSU;
- **psu_file**: dataframe containing information on each PSUs (identifier, stratum, measure of size).
- **rho**: dataframe contains a row per each stratum with the intra-class correlation coefficient both for self representing and non-self representing PSUs.

Is also possible to provide optional information about:

- **deft_start**: dataframe containing a row per each stratum with the starting values for the square root of the design effect in the stratum of each variable of interest.
- **effst**: dataframe containing a row per each stratum with the estimator effect for each variable of interest.

The functions **beat.1st** and **beat.2st** produce lists with respectively 4 and 9 items.

The **beat.1st** output contains:

1. **n**: a vector with the optimal sample size for each stratum;
2. **file_strata**: a dataframe corresponding to the input dataframe **stratif** with the n optimal sample size column added;
3. **alloc**: a dataframe with optimal (**ALLOC**), proportional (**PROP**), equal (**EQUAL**) sample size allocation;
4. **sensitivity**: a data frame with a summary of planned coefficients of variation (**Planned CV**), the expected ones under 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, as it provides a hint of the expected sample size variation for a 10% change in planned CVs.

Together with the previous outputs, the function **beat.2st** produces also:

1. **iterations**: a dataframe that for each iteration of the Bethel algorithm provides a summary with the number of PSUs (**PSU_Total**), distinguished between SR (**PSU_SR**) and NSR (**PSU_NSR**), plus the number of SSUs;
2. **planned**: a dataframe with the planned coefficients of variation for each variable in each domain.
3. **expected**: a dataframe with a summary of expected coefficients of variation under the given optimal allocation for each target variable in each domain;
4. **deft_c**: a dataframe with the design effect for each variable in each domain in each iteration. Note that **DEFT1_0 - DEFTn_0** is always equal to 1 if **deft_start** is NULL; otherwise it is equal to **deft_start**. While **DEFT1 - DEFTn** are the final design effect related to the given allocation.
5. **param_alloc**: a vector with a resume of all the parameters given for the allocation.

3.3 Sample units selection

Once the allocation for the primary and secondary sampling stage units has been defined, it is possible to use two functions for the selection of the final sampling units.

The function **select_PSU** allows the users to select the PSUs allocated in each stratum, using the Sampford method, as implemented by the **UPsampford** function of R package **sampling** (Tillé and Matei, 2021).

The input of this function is the output of the **beat.2st** function.

The output of the function is a list containing the following items:

1. **universe_PSU**: a dataframe that reports the whole universe of PSUs, with the inner strata formed for the selection;
2. **sample_PSU**: a dataframe containing the selected PSUs, with the indication, for each of them, of how many SSUs must be selected;
3. **PSU_stats**: a table containing summary information on selected PSUs.

In the last step, the selection of a sample of SSUs has to be carried out. The function **select_SSU** allows selecting a sample of SSUs from the population frame, based on the SSUs allocated to each selected PSUs.

The input datasets are two:

1. **df**: the dataframe containing the final sampling units;
2. **PSU_sampled**: the dataframe containing selected PSUs, corresponding to the second item of the output of the **select_PSU**function.

The function **select_SSU** returns a dataframe containing the selection of the **df** dataframe, enriched with information about the first stage inclusion probability, the second stage inclusion probability, the final inclusion probability (the product of the first stage and the second stage inclusion probabilities) and the design weights.

4 Illustrative examples

To illustrate how to implement workflows making use of **R2BEAT** functions, we will consider two scenarios, depending on the initial setting:

1. only the sampling frame is available, no previous rounds of the survey have been carried out;
2. together with a sampling frame containing the units of the population of reference, a previous round of the sampling survey to be planned is available;

In both cases, we assume that the sampling frame contains information on the final sampling units, together with the indication of the PSUs to which each unit belongs. In the first scenario, a stricter condition on the information content of the sampling frame must hold: values of the sample target surveys (or of their proxy correlated variables) must be available for each unit in the frame. This can be accomplished by considering a previous census, or by imputing values using predictive models. In the following paragraphs, we will show only a subset of the code necessary to produce the final results, the relevant part of it[1].

4.1 Scenario 1 workflow

In this scenario, it is assumed that a sampling frame is available. We consider a frame (**pop.RData**), containing 2,258,507 units:

```
'data.frame':   2258507 obs. of  13 variables:
$ region       : Factor w/ 3 levels "north","center",..: 1 1 1 1 1 1 1 1 1 1 ...
$ province     : Factor w/ 6 levels "north_1","north_2",..: 1 1 1 1 1 1 1 1 1 1 ...
$ municipality : num  1 1 1 1 1 1 1 1 1 1 ...
$ id_hh       : Factor w/ 963018 levels "H1","H10","H100",..: 1 1 1 2 3 3 3 1114 1114 ...
$ id_ind       : int  1 2 3 4 5 6 7 8 9 10 ...
$ stratum      : Factor w/ 24 levels "1000","2000",..: 12 12 12 12 12 12 12 12 12 12 ...
$ stratum_label: chr  "north_1_6" "north_1_6" "north_1_6" "north_1_6" ...
$ sex          : int  1 2 1 2 1 1 2 2 1 1 ...
$ cl_age       : Factor w/ 8 levels "(0,14]","(14,24]",..: 3 7 8 5 4 6 6 4 4 1 ...
$ active       : num  1 1 0 1 1 1 1 1 1 0 ...
$ income_hh    : num  30488 30488 30488 21756 29871 ...
$ unemployed   : num  0 0 0 0 0 0 0 0 0 0 ...
$ inactive     : num  0 0 1 0 0 0 0 0 0 1 ...
```

covering a (synthetic) population of reference, with basic information (geographical and demographic variables:

- region: the NUTS2 identifier;
- province: the NUTS3 identifier;
- municipality: identifier of the municipality, that plays the role of the PSU identifier;
- id_hh: the household identifier;
- id_ind: the individual identifier;
- stratum and stratum_label: identifier of the initial strata (provinces);
- sex and cl_age: demographic information on individuals.

together with information that is related to the sampling survey we want to design:

[1] In order to reproduce the processing related to these examples, datasets and R scripts are downloadable from the link https://zenodo.org/records/10183968.

- **active**, **inactive**, **unemployed**: binary variables indicating the occupation status of the individual;
- **income_hh**: household income.

We suppose that the values of these variables have been made available by a different source (for instance a census) or by predicting them with a model-based approach. In any case the uncertainty related to these values should be taken into account, by correctly evaluating the anticipated variance related to the models used for the predictions when producing the **strata** dataset (Baillargeon and Rivest, 2011, p. 59).

Anyway, in the following, we will not consider this issue, as we want only to illustrate how it is possible to automatically derive all the inputs required by the next steps.

Step 1: preparation of the inputs for the optimal sample design

The function **prepareInputToAllocation1** allows preparing all the inputs required by the optimal allocation step under this first scenario. This function requires the attribution of values to the following parameters: **samp_frame**, **id_PSU**, **id_SSU**, **strata_var**, **target_vars**, **deff_var**, **domain_var**, **delta** (average dimension of the SSU in terms of elementary survey units), **minimum** (minimum number of SSUs to be interviewed in each selected PSU).

About the values of these parameters, the choices are almost always driven by the content and structure of the sampling frame, except for **minimum**. To choose a suitable value for this parameter, the function **sensitivity_min_SSU** allows performing a sensitivity analysis, showing how the first and second stage sample sizes vary by varying their values:

```
> sens_min_SSU <- sensitivity_min_SSU (
+             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",
+             delta=1,
+             deff_sugg=1.5,
+             min=30,
+             max=80)
```

This function calculates 11 different pairs of values for the number of PSUs and SSU as resulting from the allocation step, starting with the value '30' assigned to the parameter **minimum**, ending with the value '80'. The results are reported in Figure 1.

On the basis of the results of the sensitivity analysis, we can set the minimum (for instance 50) in the **prepareInputToAllocation1** function:

```
> inp <- prepareInputToAllocation1(
+             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",
+             delta = delta,
+             minimum = 50,
+            deff_sugg = 1.5)
 Computations are being done on population data
 Number of strata:  24
 ... of which with only one unit:  0
```

The output of this function (**inp**) is a list composed by the following dataframes: **strata**, **deff**, **effst**, **rho**, **psu_file**, **des_file**. These will be the inputs for the optimal allocation step (with the exception of the **deff**), which is produced only for documentation).

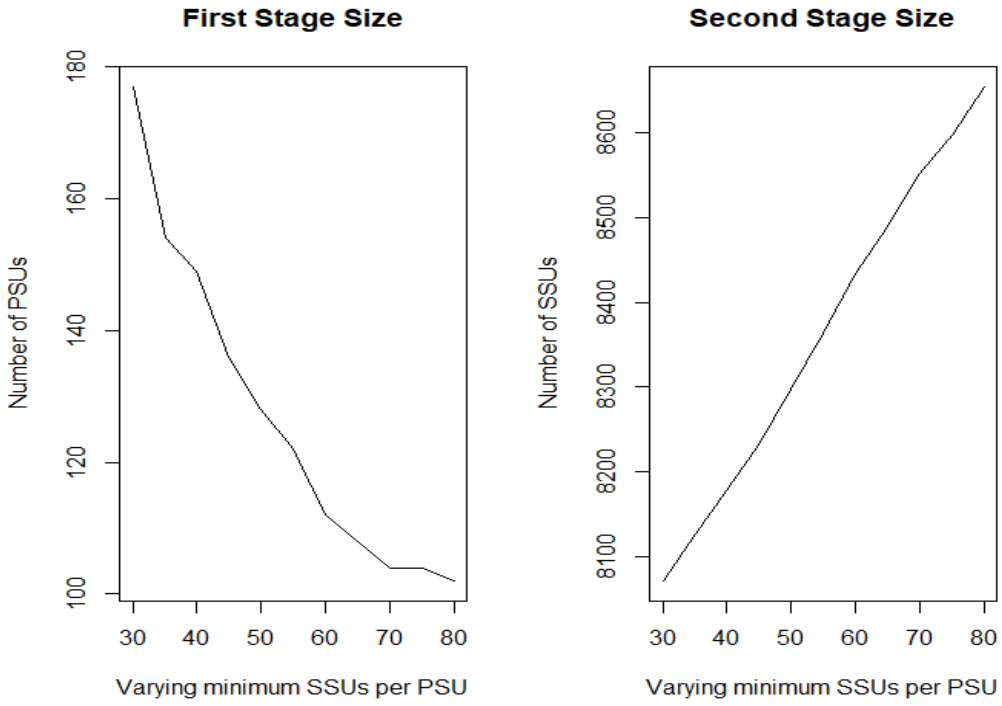

Figure 1: Sensitivity analysis for **minimum** parameter.

Step 2: optimization of PSUs and SSUs allocation

It is now possible to execute the optimization step of the sample design.

First of all, we define the set of precision constraints on the target variables:

```
DOM  CV1  CV2  CV3  CV4
1 DOM1 0.02 0.03 0.03 0.05
2 DOM2 0.03 0.06 0.06 0.08
```

We interpret the values of the CVs in this way: the maximum expected coefficient of variation for the first target variable (**household income**) is 2% at the national level and 3% at the regional level; for **active** and **inactive** the expected maximum values of CV is 3% at the national level and 6% at the regional level; finally, for **unemployed** it is 5% at the national level and 8% at the regional level.

The optimization step is performed by executing the **beat.2s** function:

```
> inp1$desfile$MINIMUM <- 50
> alloc1 <- beat.2st(stratif = inp1$strata,
+                    errors = cv,
+                    des_file = inp1$des_file,
+                    psu_file = inp1$psu_file,
+                    rho = inp1$rho,
+                    deft_start = NULL,
+                    effst = inp1$effst,
+                    minPSUstrat = 2,
+                    minnumstrat = 50
+                    )
iterations PSU_SR PSU NSR PSU Total  SSU
1          0      0       0         0 7887
2          1     31     104       135 8328
3          2     39     104       143 8318
4          3     38     104       142 8321
```

This design is characterized by 142 PSUs (of which 38 self-representative, SR, and 104 non self-representative, NSR) and 8,321 SSUs.

Step 3: selection of PSUs and SSUs

We can now proceed in selecting the PSUs:

```
> sample_1st <- select_PSU(alloc, type="ALLOC", pps=TRUE, plot=TRUE)
> sample_1st$PSU_stats
    STRATUM PSU PSU_SR PSU_NSR  SSU SSU_SR SSU_NSR
1      1000   2      2       0  286    286       0
2      2000   9      3       6  452    152     300
3      3000   4      0       4  200      0     200
...
23    23000   4      0       4  200      0     200
24    24000   2      0       2  100      0     100
25    Total 142     38     104 9421   4221    5200
```

A discrepancy can be noted between the number of SSUs determined by the allocation step and the one produced by the selection of PSUs. This is because the selection of PSUs controls that the minimum number of SSUs to be allocated in each selected PSU is compliant with the minimum, in our case equal to 50: if not, this minimum is assigned. This is why the total number of SSUs increases from 8,320 to 9,421.

Selected PSUs are contained in the **sample_PSU** element of the output list:

```
> head(sample_1st$sample_PSU)
  PSU_ID STRATUM stratum SR nSR PSU_final_sample_unit Pik weight_1st weight_2st   weight
1    330    1000  1000-1  1   0                  207   1          1   706.0966 706.0966
2    309    1000  1000-2  1   0                   72   1          1   706.1806 706.1806
3     51   10000 10000-0  1   0                  171   1          1   196.8480 196.8480
4     11   10000 10000-1  1   0                   96   1          1   196.9688 196.9688
5     40   10000 10000-2  1   0                   79   1          1   197.9494 197.9494
6     13   10000 10000-3  1   0                   72   1          1   198.3750 198.3750
```

With this input, we can proceed to select the sample of final units. The distribution of PSUs and SSUs in the different strata is reported in Figure 2.

```
> PSU_sampled <- sample_1st$sample_PSU
> samp <- select_SSU(df=pop,
+                    PSU_code="municipality",
+                    SSU_code="id_ind",
+                    PSU_sampled)

--------------------------------
Total PSUs =  142
Total SSUs =  9421
--------------------------------
```

Step 4: verify compliance with precision constraints

The function **eval_2stage** allows verifying the compliance of the two-stage sample design to the set of precision constraints, by selecting a given number of different samples (in our case, 500) from the sampling frame, producing the estimates for each sample, and calculating over them the coefficients of variation for each target estimate.

We apply twice the function, first for the national level:

```
> # Domain level = national
> domain_var <- "one"
> set.seed(1234)
> eval11 <- eval_2stage(df,
+                       PSU_code,
+                       SSU_code,
+                       domain_var,
+                       target_vars,
+                       sample_1st$sample_PSU,
```

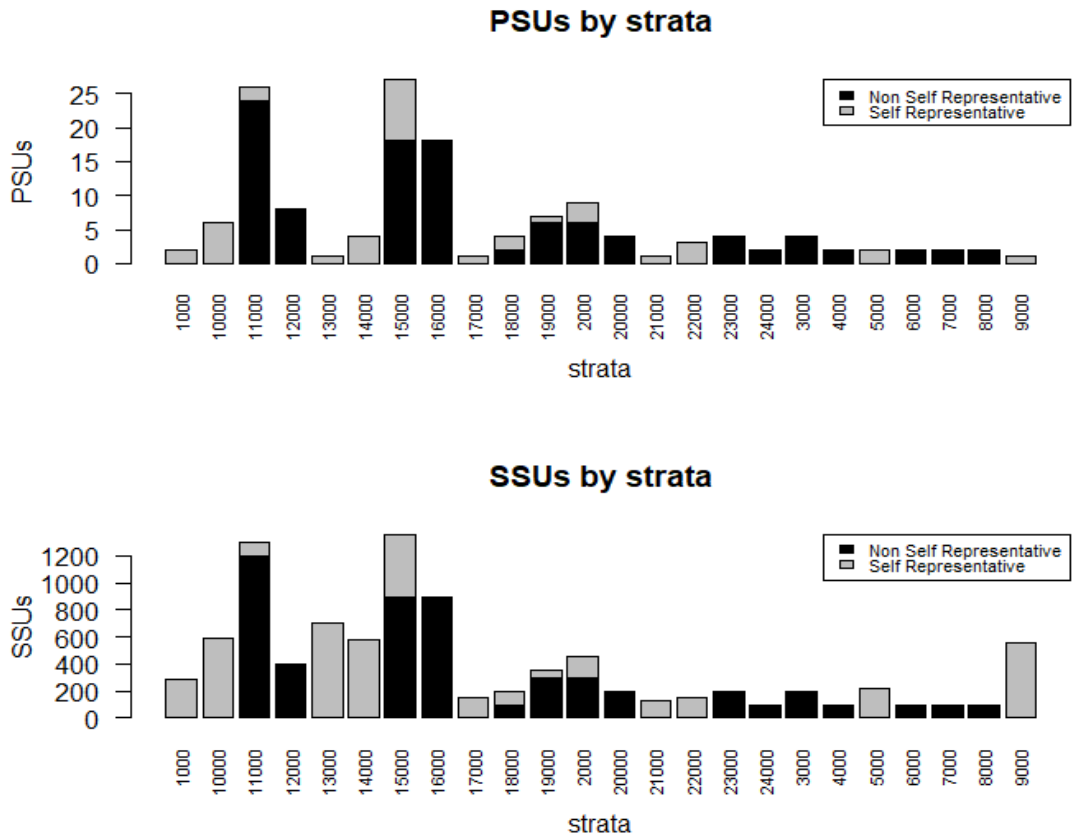

Figure 2: Allocation of PSUs and SSUs (scenario 1).

```
 +                    nsampl=100,
 +                    writeFiles=FALSE,
 +                    progress=FALSE)
 > eval11$coeff_var
     CV1    CV2    CV3    CV4  dom
1 0.0093 0.0099  0.025 0.0356 DOM1
```

then, at the regional level:

```
 > # Domain level = regional
 > domain_var <- "region"
 > set.seed(1234)
 > eval12 <- eval_2stage(df,
 +                    PSU_code,
 +                    SSU_code,
 +                    domain_var,
 +                    target_vars,
 +                    sample_1st$sample_PSU,
 +                    nsampl=100,
 +                    writeFiles=FALSE,
 +                    progress=FALSE)
 > eval12$coeff_var
     CV1    CV2    CV3    CV4  dom
1 0.0105 0.0061 0.0224 0.0702 DOM1
2 0.0241 0.0171 0.0452 0.0652 DOM2
3 0.0281 0.0279 0.0533 0.0446 DOM3
```

We recall that the precision constraints had been set equal to 2% for the first variable, 3% for the second and third, and 5% for the fourth, at national level; and respectively to 3% and 6% and 8% at regional level. We can see that the computed CVs are all compliant.

4.2 Scenario 2 workflow

Together with the availability of a sampling frame, containing the same information presented in the previous scenario, we assume also the availability of at least one previous round of the survey. For sake of simplicity, we assume that the previous round sample is the same selected in scenario 1. We assume also that the values of the four target variables are the observed ones after the data collection. Having set the above conditions, the main difference with scenario 1 is that, instead of choosing in a somewhat arbitrarily way the values of the inputs required by the optimal allocation step, we can derive them directly from the collected survey data.

Step 1: processing and analysis of survey data

In this step, we proceed to perform the usual phases of calibration and production of the estimates. In doing that, we make use of the R package **ReGenesees**.

First, we describe the sample design:

```
> ## Sample design description
> sample$stratum_2 <- as.factor(sample$stratum_2)  # stratum as factor, required by the next function
> sample.des <- e.svydesign(sample,
+                           ids= ~ municipality + id_hh,
+                           strata = ~ stratum_2,
+                          self.rep.str = ~ SR,
+                           weights = ~ weight,
+                           check.data = TRUE)
```

obtaining the **sample.des** object. Then we proceed with the calibration step:

```
> ## 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(0.676, 1.279),
+                           aggregate.stage = 2,
+                           force = FALSE)
```

obtaining the **sample.cal** object.

These two objects are what is needed to obtain, in an automated way, all the inputs required by the optimization step.

Step 2: preparation of the inputs for the optimal sample design

The preparation of all the inputs required by the optimization step is a straightforward operation by using the **prepareInputToAllocation2** function:

```
> inp <- prepareInputToAllocation2(
+         samp_frame = pop,             # sampling frame
+         RGdes = sample.des,           # ReGenesees design object
+         RGcal = sample.cal,           # ReGenesees calibrated object
+         id_PSU = "municipality",      # identification variable of PSUs
+         id_SSU = "id_hh",             # identification variable of SSUs
+         strata_vars = "stratum",      # strata variables
+        target_vars = c("income_hh","active","inactive","unemployed"), # target variables
+         deff_vars = "stratum",        # deff variables
+         domain_vars = "region",         # domain variables
+        delta = 1,                   # Average number of SSUs for each selection unit
+         minimum= 50)               # Minimum number of SSUs to be selected in each PSU
```

The configuration of the output is the same as that seen in scenario 1 for the function **prepareInputToAllocation1**.

Step 3: optimization of PSUs and SSUs allocation

The optimal allocation of PSUs and SSUs is obtained in the same way as in the first scenario:

```
 > set.seed(1234)
 > inp2$des_file$MINIMUM <- 50
 > alloc2 <- beat.2st(stratif = inp2$strata,
 +                    errors = cv,
 +                    des_file = inp2$des_file,
 +                    psu_file = inp2$psu_file,
 +                    rho = inp2$rho,
 +                    deft_start = NULL,
 +                    effst = inp2$effst,
 +                    minnumstrat = 2,
 +                    minPSUstrat = 2)
   iterations PSU_SR PSU NSR PSU Total  SSU
1          0      0       0         0 9546
2          1     71      92       163 8475
3          2     40     108       148 8406
4          3     38     108       146 8404
```

Step 4: selection of PSUs and SSUs

The selection of first and second stage units proceeds in exactly the same way as in scenario 1, first selecting the PSUs, and then the SSUs.

```
 > sample_1st <- select_PSU(alloc2, type="ALLOC", pps=TRUE)
 > sample_1st$PSU_stats
   STRATUM PSU PSU_SR PSU_NSR  SSU SSU_SR SSU_NSR
1     1000   2      2       0  276    276       0
2     2000  10      6       4  517    317     200
3     3000   4      0       4  200      0     200
...
24   24000   2      0       2  100      0     100
25   Total 146     38     108 9580   4180    5400
 >
 > samp <- select_SSU(df=pop,
 +                    PSU_code="municipality",
 +                    SSU_code="id_ind",
 +                    PSU_sampled=sample_1st$sample_PSU,
 +                    verbose=TRUE)
--------------------------------
Total PSUs =  144
Total SSUs =  9465
--------------------------------
```

The distribution of PSUs and SSUs in the different strata is reported in Figure 3. It can be seen that the relative distribution of both units in the strata is quite similar to the one obtained in scenario 1.

Step 5: verify the compliance to precision constraints

As in the previous scenario, the final check consists in verifying the compliance of the optimized design to the precision constraints.

We, therefore, apply the function **eval_2stage**, first for the national level:

```
 > # Domain level = national
 > domain_var <- "one"
 > eval21 <- eval_2stage(df,
 +                       PSU_code,
```

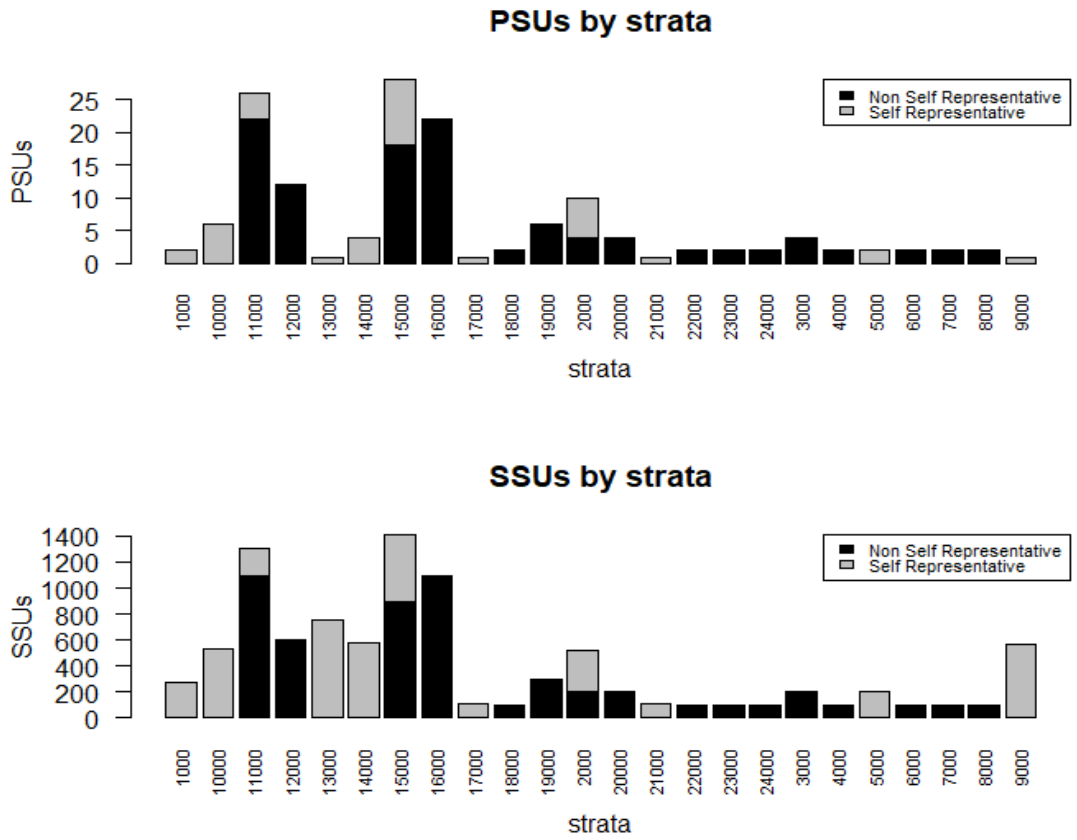

Figure 3: Allocation of PSUs and SSUs (scenario 2).

```
 +                    SSU_code,
 +                    domain_var,
 +                    target_vars,
 +                    PSU_sampled=sample_1st$sample_PSU,
 +                    nsampl=100)
 > eval21$coeff_var
     CV1    CV2    CV3    CV4  dom
1 0.0102 0.0095 0.0232 0.0354 DOM1
```

then, at regional level:

```
 > # Domain level = regional
 > domain_var <- "region"
 > eval22 <- eval_2stage(df,
 +                     PSU_code,
 +                     SSU_code,
 +                     domain_var,
 +                     target_vars,
 +                     PSU_sampled=sample_1st$sample_PSU,
 +                     nsampl=100)
 > eval22$coeff_var
     CV1    CV2    CV3    CV4  dom
1 0.0108 0.0067 0.0243 0.0824 DOM1
2 0.0240 0.0202 0.0514 0.0671 DOM2
3 0.0275 0.0353 0.0633 0.0417 DOM3
```

In this case, the expected precision is slightly higher than the precision constraints in two cases: the fourth variable in domain 1, and the third variable in domain 3. This can be due to the fact that in this scenario the variance of the target variables in strata (on the basis of which the total sample

size and the allocation are determined) is not derived from the sampling frame (as in scenario 1), but estimated from a previous round of the survey.

5 Comparison with other software

To evaluate the performance of R2BEAT, in this section we compare it to the other two R packages, i.e.:

1. the package **PracTools** (Valliant et al., 2020) implements many of the procedures described in Valliant et al. (2015), including those regarding the design of multistage samples;
2. the package **samplesize4surveys** (Rojas, 2020) allows to calculate the sample size for complex surveys.

First, we briefly illustrate, for both packages, the functions covering the two-stage sampling design, then we apply them to the same case seen in scenario 1, finally comparing the obtained results [2].

5.1 R package PracTools

Valliant et al. (2015) describe (pages 231-234) a method for the optimal allocation of two-stage sampling when numbers of sample PSUs and elements per PSU are adjustable (which is our case).

This method is implemented in the R function **clusOpt2** in the **PracTools** package. This function computes the number of PSUs and the number of final units for each PSU for a two-stage sample which uses *srs* at each stage or probability proportional to size with replacement (*ppswr*) at the first stage and *srs* at the second.

This function requires the indication of a number of parameters, among which:

- C1: unit cost per PSU
- C2: unit cost per SSU
- delta: homogeneity measure
- unit.rv: unit relvariance
- k: ratio of B2+W2 to unit relvariance
- CV0: target CV
- tot.cost: total budget for variable costs
- cal.sw: indicates if the optimization has to be run for a fixed total budget, or for the target CV0

The function **BW2stagePPS** computes the population values of B2, W2, and delta whose meaning is explained in Valliant et al. (2015) (page 222).

The method is univariate: the optimization can be performed by indicating only one variable. The whole code required for the case described in scenario 1 is given here:

```
> load("pop.RData")
> library(PracTools)
> # Probabilities of inclusion (I stage)
> pp <- as.numeric(table(pop$municipality))/nrow(pop)
> # variable income_hh
> bw <- BW2stagePPS(pop$income_hh, pp, psuID=pop$municipality)
> bw
        B2          W2 unit relvar       B2+W2           k       delta
0.04075893  0.79538674  0.83601766  0.83614567  1.00015312  0.04874621
> des <- clusOpt2(C1=130,
+                 C2=1,
+                 delta=bw[6],
+                 unit.rv=bw[3],
+                 k=bw[5],
+                 CV0=0.02,
+                 tot.cost=NULL,
+                 cal.sw=2)
> des
C1 = 130
```

[2] In order to reproduce the processing related to the evaluation of the different software, datasets and R scripts are downloadable from the link https://zenodo.org/records/10184220

```
C2 = 1
delta = 0.04874621
unit relvar = 0.8360177
k = 1.000153
cost = 25499.72
m.opt = 141.4
n.opt = 50.4
CV = 0.02
> sample_size <- des$m.opt*des$n.opt
> sample_size
7126.56
```

In running the function, we have indicated that the optimization step was to be carried out having a target CV of 2% for the variable **income_hh**. As there is no way to directly indicate a desired minimum number of SSUs per PSU, we managed to obtain the desired value of 50 by indicating a couple of values 130 and 1 respectively for C1 and C2. As a result, the number of PSUs is 141 and the number of SSUs is 7,127.

5.2 R package samplesize4surveys

This package offers two functions to compute a grid of possible sample sizes for estimating single means (**ss2s4m**) or single proportions (**ss2s4p**) under two-stage sampling designs.

The required parameters are the following:

- N: the population size
- mu: the value of the estimated mean of a variable of interest
- sigma: the value of the estimated standard deviation of a variable of interest
- conf: the statistical confidence
- delta: the maximum relative margin of error that can be allowed for the estimation
- M: number of clusters in the population
- to: (integer) maximum number of final units to be selected per cluster
- rho: the intraclass correlation coefficient

Here is the code we used in the case of the target variable **income_hh**:

```
> load("pop.RData")
> PSU <- length(unique(pop$municipality))
> pop_strata <- as.numeric(table(pop$stratum))
> rho <- 0.04875369 # value taken from scenario 1 analysis
> ss2s4m(N = nrow(pop),
+        mu = mean(pop$income_hh),
+        sigma = sd(pop$income_hh),
+        delta = 0.02 * 1.96,
+        M = PSU,
+        to = 50,
+        rho = sum(rho$RHO_NAR1*pop_strata) / sum(pop_strata))
50 3.388931 142 50 7061
```

we obtain a design characterized by a total sample size of 7,061, with 142 PSUs.

Concerning the way we indicated the value of the parameter **rho**, we made use of the value of the intra-class correlation coefficient computed in scenario 1 by **R2BEAT**, not considering domains and strata.

In order to compare the 2% precision constraint expressed in terms of coefficient of variation, as the package requires the margin of error, we multiply the value of the CV by a z-value equal to 1.96, to obtain the ratio between the semi-width of the confidence interval and the estimate of the mean of the parameter.

The use of the function **ss2s4p**, applicable for the other three variables, is practically the same.

5.3 Comparison of results

We refer to the scenario 1 setting.

We consider the same precision levels for the four variables for the unique domain, set respectively to 2%, 3%, 3% and 5%.

We apply another constraint for all three packages, that is, we want to select a minimum number of final units in each PSU, set to 50.

There is no problem in doing that for package **samplesize4surveys**, by setting the parameter **to** equal to 50: the last value of the final grid is the result we want. Moreover, there is no loss in the optimality of the solution in doing that, because the sample sizes obtained for further values are increasingly higher.

As for **PractTools**, it is more complicated because there is no direct way to set this constraint. In any case, we manage to do that, by varying the value of C1 (leaving C2 equal to 1) until we find the solution with the nearest value of **n.opt** to 50.

A final consideration regarding the application of **R2BEAT**: in this setting, to be comparable with the other packages (that are univariate and mono-domain), it has been applied in a simplified way, that is, one variable per time (univariate), and no different domains and strata in the sampling frame. By so doing, **R2BEAT** yields obviously different results from those seen in scenario 1.

Table 1: Two-stage sample design obtained by different packages.

	PracTools		R2BEAT		samplesize4surveys	
Variable	PSUs	SSUs	PSUs	SSUs	PSUs	SSUs
active	49	2459	37	2030	49	2436
inactive	90	4395	68	4338	88	4391
income_hh	141	7127	79	5140	142	7061
unemployed	406	19956	149	10884	402	20058

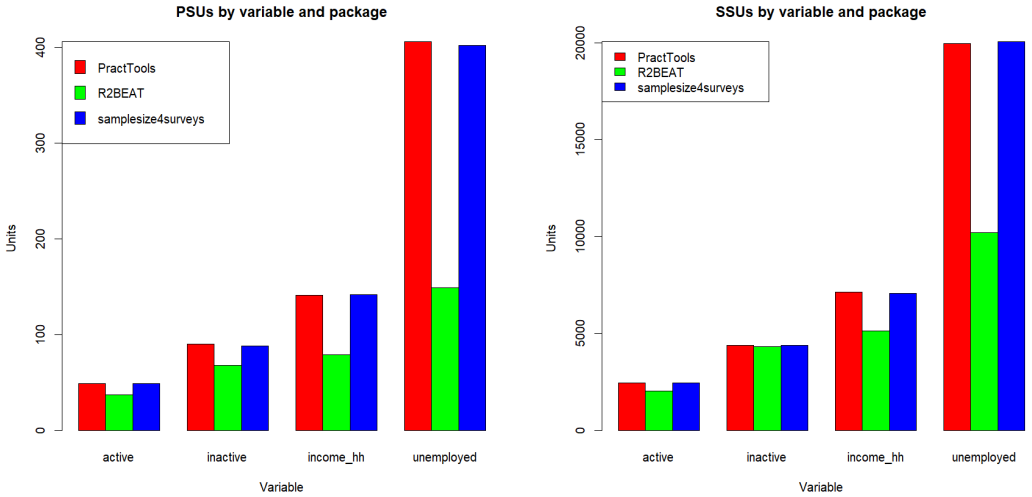

Figure 4: Sample sizes by packages.

In Table 1 and in Figure 4 are reported the results obtained by the three packages. For checking the reliability of the results, also a simulation has been performed to prove that **R2BEAT** provides sample sizes that always allow to obtain CVs below the planned ones.

Going in-depth with the comparison among the packages, it can be seen that **PracTools** and **samplesize4surveys** have similar behaviors and provide very close results, while **R2BEAT** always provide samples with smaller sample sizes, both in terms of PSUs and SSUs.

The differences related to the SSUs are smaller but there are two extremes. For “inactive" all three packages yield almost the same results in terms of SSUs, while for “unemployed” **R2BEAT** outperforms the other two competitors almost halving their sample sizes.

Juxtaposing Table 1 and 2, it is possible to see that the differences can be related to the intraclass correlation coefficient, ρ. The differences are maximum in the case of the variable “unemployed”

which has the higher ρ, and is almost negligible in the case of the variable "inactive" for which ρ is close to 0.

Focusing on the different uses that the three packages make of ρ, the differences in the final allocation results arise. In **PracTools** and **samplesize4surveys**, ρ is only involved in the direct calculation of the optimal number of both PSUs and SSUs, once and for all. Furthermore, in **samplesize4surveys**, ρ is used only to calculate the $deff$, in turn, used to inflate the variance of the estimate but, also in this case, once and for all. While, in **R2BEAT**, ρ is involved in the partition of the PSUs in self-representative (SR) and non-self-representative (NSR).

This partition, updated every iteration, takes ρ into account each time. In fact, the iterative procedure - and of course the use of ρ in each iteration - makes the results more stable and explores, among all the suitable solutions, the more efficient in terms of PSUs and SSUs.

Summarising, the higher the value of the ρ, the higher the efficiency of the R2BEAT algorithm that makes extensive use of the ρ values.

Table 2: Intraclass correlation coefficient (ρ) for active, inactive, income, and unemployed in the whole population.

Variable	ρ
active	0.0656
inactive	0.0021
income_hh	0.0488
unemployed	0.1263.

6 How to manage the total non-response

Sample size and allocation in strata provided by **R2BEAT** ensure compliance with the precision constraints. Unfortunately, in practice, the planned sample size is affected by total non-response which reduces it and, consequently, weakens the precision and efficiency of the estimates. Therefore, it is important to consider the total non-response when planning sample surveys and to implement strategies to preserve the precision of the estimates.

The most common methods to carry out this task at the planning stage are *oversampling* and *substitutions*.

Oversampling is a technique used to increase the sample size of a population in a survey according to the total non-response rate in the population strata (nrr_h, with $0 \leq nrr_h \leq 1$ and $h = 1, \ldots, H$) for recovering, at the end of the data collection phase, the desired sample size and the expected precision requirements. These rates are usually derived from previous survey occasions, carried out with the same data collection technique and under similar conditions, or on reliable assumptions made by those who design the survey. For embedding this procedure in **R2BEAT**, before performing the selection of the SSUs, it is sufficient to multiply the column **PSU_final_sample_unit** of the **data.frame sample_PSU** provided as the output of the function **select_PSU** by $1 + nrr_h$. If the response rates are not available at the strata level but at a higher domain level (for instance, for domains that are aggregations of strata), it is possible to assume a constant non-response rate for the strata belonging to the same domain. Of course, the more precise the knowledge of response rates, the more accurate the oversampling procedure adopted will be.

Instead, substitutions involve replacing non-responding units with other ones which have been "a priori" selected. Sometimes even more than one substitute for a single SSU is singled out. The selection of substitute units can be carried out in different ways (see, for a brief but complete review Lynn, 2004). If a simple random substitution of the SSUs, before performing the selection of the SSU, it is sufficient to multiply the column **PSU_final_sample_unit** of the **data.frame sample_PSU** provided as the output of the function **select_PSU** by the chosen number of substitutes. Otherwise, if a non-random substitution or a stratified substitution, unlike in the previous case, the selection of SSUs requires additional information on the SSUs and, therefore, the use of ad hoc solutions.

However, it is important to point out that substitutions and oversampling aim to preserve the sample size inflating it to withstand the non-response. However, it could be anyway not enough to avoid total non-response bias side effects on the final estimates. Therefore, non-response treatment methods must be applied after the data collection phase (see, e.g., Särndal and Lundström, 2005; Little and Rubin, 2019).

7 Concluding remarks

Methodology, completeness, and efficiency can be considered the main strengths of the R package R2BEAT.

The methodology based on the application of the Bethel algorithm ensures that the resulting sample designs are compliant with the precision constraints set on the target estimates of a given survey.

The completeness can be considered both:

- in terms of applicability: R2BEAT applies to stratified, multipurpose and multidomain, one-stage and two-stage sampling surveys;
- in terms of coverage of the user needs: the package covers all the steps of a complex sample design, from the setting of precision constraints, the determination of the total sample size and of the allocation in the strata and the selection of the sampling units, distinctly according to the stage of selection.

Moreover, a set of functions generates the input required by the optimization step starting from the availability of a sampling frame and/or a previous round of the survey, thus allowing the users to fully harness all the available information with minimal effort. Other functions help them to perform analysis and checks on the obtained allocations and the selected sample.

Finally, we demonstrated the efficiency of the package by comparing it with two other competitor packages in terms of the results obtained in a case study: on equal errors, the sample size determined by R2BEAT is always lower, in terms of both Primary and Secondary Stage Units (PSUs and SSUs).

In future work, considering that a limitation of the package is its applicability to only mean parameters, the extension to more complex parameters will be investigated.

References

S. Baillargeon and L.-P. Rivest. The construction of stratified designs in R with the package stratification. *Survey Methodology*, 37(1):53–65, 2011. [p201]

G. Barcaroli. SamplingStrata: An R package for the optimization of stratified sampling. *Journal of Statistical Software*, 61(4):1–24, 2014. [p191]

G. Barcaroli, T. Buglielli, and C. D. Vitiis. *MAUSS-R: Multivariate Allocation of Units in Sampling Surveys*, 2020. URL https://www.istat.it/en/methods-and-tools/methods-and-it-tools/design/design-tools/mauss-r. R package version 2.4. [p191]

J. W. Bethel. Sample allocation in multivariate surveys. *Survey methodology*, 15(1):47–57, 1989. [p193]

J. Breidaks, M. Liberts, and J. Jukams. *surveyplanning: Survey planning tools*. Riga, Latvia, 2020. URL https://csblatvia.github.io/surveyplanning/. R package version 4.0. [p191]

E. Bueno. *optimStrat: Choosing the Sample Strategy*, 2020. URL https://CRAN.R-project.org/package=optimStrat. R package version 2.3. [p191]

G. Cicchitelli, A. Herzel, and G. E. Montanari. *Il campionamento statistico*. Il mulino, Bologna, 1992. [p194]

W. G. Cochran. *Sampling techniques*. John Wiley & Sons, 1977. [p191, 194]

J.-C. Deville and C.-E. Särndal. Calibration estimators in survey sampling. *Journal of the American statistical Association*, 87(418):376–382, 1992. [p195]

P. D. Falorsi, M. Ballin, C. De Vitiis, and G. Scepi. Principi e metodi del software generalizzato per la definizione del disegno di campionamento nelle indagini sulle imprese condotte dall'istat. *Statistica Applicata*, 10(2):235–257, 1998. [p195]

M. H. Hansen, W. N. Hurwitz, and W. G. Madow. Sample survey methods and theory. Vol. I and II, Methods and applications. 1953. [p194]

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952. [p192]

L. Kish. *Survey sampling*. John Wiley & Sons, Inc., New York, 1965. [p191]

L. Kish. Optima and proxima in linear sample designs. *Journal of the Royal Statistical Society: Series A (General)*, 139(1):80–95, 1976. [p193]

R. J. Little and D. B. Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019. [p211]

P. Lynn. The use of substitution in surveys. *The Survey Statistician*, 49:14–16, 2004. [p211]

J. Neyman. 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, 1934. [p192]

H. A. G. Rojas. *Estrategias de muestreo: diseño de encuestas y estimación de parámetros*. Ediciones de la U, 2016. [p195]

H. A. G. Rojas. *samplesize4surveys: Sample Size Calculations for Complex Surveys*, 2020. URL https://CRAN.R-project.org/package=samplesize4surveys. R package version 4.1.1. [p191, 208]

C.-E. Särndal and S. Lundström. *Estimation in surveys with nonresponse*. John Wiley & Sons, 2005. [p211]

C.-E. Särndal, B. Swensson, and J. Wretman. *Model assisted survey sampling*. Springer Science & Business Media, 2003. [p194]

Y. Tillé and A. Matei. *sampling: Survey Sampling*, 2021. URL https://CRAN.R-project.org/package=sampling. R package version 2.9. [p199]

A. Tschprow. On the two different aspects of the representative method: the method of stratified son the mathematical expectation of the moments of frequency distributions in the case of correlated observation sampling and the method of purposive selection. *Metron*, 2:646–683, 1923. [p192]

R. Valliant, J. A. Dever, and F. Kreute. *Practical Tools for Designing and Weighting Survey Samples*. Springer, 2015. [p208]

R. Valliant, J. A. Dever, and F. Kreuter. *PracTools: Tools for Designing and Weighting Survey Samples*, 2020. URL https://CRAN.R-project.org/package=PracTools. R package version 1.2.2. [p191, 208]

J. R. Waters and A. J. Chester. Optimal allocation in multivariate, two-stage sampling designs. *The American Statistician*, 41(1):46–50, 1987. [p195]

D. Zardetto. ReGenesees: An advanced R system for calibration, estimation and sampling error assessment in complex sample surveys. *Journal of Official Statistics*, 31(2):177–203, 2015. [p198]

Giulio Barcaroli
Independent consultant
via Monte delle Gioie 29 - 00199 Roma
Italy
gbarcaroli@gmail.com

Andrea Fasulo
Italian National Institute of Statistics
via Cesare Balbo 16 - 00184 Roma
Italy
andrea.fasulo@istat.it

Alessio Guandalini
Italian National Institute of Statistics
via Cesare Balbo 16 - 00184 Roma
Italy
alessio.guandalini@istat.it

Marco D. Terribili
Italian National Institute of Statistics
via Cesare Balbo 16 - 00184 Roma
Italy
marco.terribili@istat.it