Introduction to SamplingStrata R package

Optimal stratification of a sampling frame and allocation and selection of units

Giulio Barcaroli

April 2011

Introduction

So far, in stratified random sampling the problem of determining the optimal size and allocation of units in strata has been solved by considering the stratification of population as given. Conversely, the definition of an optimal stratification of a sampling frame for a given survey has been investigated without choosing, as objective function, the sampling size required to satisfy given precision constraints on parameters of interest of the survey. This package allows the determination of the best stratification of a target population, the one that ensures the minimum sample size (or the minimum field and interviewing costs) so to satisfy precision constraints in a multivariate and multidomain case. The underlying algorithm is based on a non deterministic evolutionary approach, making use of the genetic algorithm paradigm.

1. Problem definition

Let us suppose we need to carry out a sample survey, having a complete frame containing information on the target population (identifiers plus auxiliary information). If our sample design is a stratified one, we need to choose how to form strata in the population, in order to get the maximum advantage by the available auxiliary information. In other words, we have to decide in which way to combine the values of the auxiliary variables (from now on, the "X" variables) in order to determine a new variable, called "stratum". To do so, we have to take into consideration the target variables of our sample surveys (from now on, the "Y" variables): if, to form strata, we choose the Xs variables most correlated to the Ys, the efficiency of the samples drawn by the resulting stratified frame may be greatly increased. In order to handle the whole auxiliary information in a homogenous way, we have to reduce continuous data to categorical (by mean of a k-means clustering technique, for example). Then, for every set of candidate auxiliary variables Xs, we have to decide (i) what variables to consider as active variables in strata determination, and (ii) for each active variable, what set of values (in general, what aggregation of atomic values) have to be considered. Every combination of values of each active variable determine a particular stratification of the target population, i.e. a possible solution to the problem of "best" stratification. Here, by *best* stratification, we mean the stratification that ensures the minimum sample size, sufficient to comply a set of precision constraints on the accuracy of the estimates of the survey target variables Ys (constraints expressed as maximum allowable sampling variance on estimates in different domains of interest).

Therefore, the validity of a particular stratification can be measured by the associated minimum size of a sample, whose estimates are expected to comply given accuracy levels. This minimum size can be determined by applying the Bethel algorithm, with its Chromy implementation. In general, the number of possible alternative stratifications for a given population may be very high, depending on the number of variables and on the number of their values, and in these cases it is not possible to enumerate them in order to assess the best one. A very convenient solution to this, is the adoption of the *evolutionary approach*, consisting in applying a genetic algorithm that may converge towards a near-optimal solution after a finite number of iterations.

2. Procedural steps (1-9)

The optimisation of the sampling design starts from making the sampling frame available, defining the target estimates of the survey and establishing the precision constraints on them, determining the best stratification and the optimal allocation, ending with the selection of the sample.

Formalising, these are the required steps:

1. analysis of the frame data: identification of available auxiliary information;
2. manipulation of auxiliary information: in case auxiliary variables are of the continuous type, they must be reduced in a categorical form;
3. construction of *atomic* strata: on the basis of the categorical auxiliary variables available in the sampling frame, a set of strata can be constructed by calculating the Cartesian product of the modalities of auxiliary variables;
4. association of information related to target variable to each atomic stratum: in order to optimise both strata and allocation of sampling units in strata, we need information on the distributions of the target variables (means and standard deviations);
5. choice of the precision constraints for each target estimate, differentiated by domain;
6. optimization of stratification and determination of required sample size and allocation in order to satisfy precision constraints on target estimates;
7. analysis of the resulting optimized strata;
8. association of new labels to sampling frame units, each of them indicating the new strata resulting by the optimal aggregation of the atomic strata;
9. selection of units from the sampling frame with a stratified sampling random sample selection scheme.

In the following, we will illustrate each step starting from a real sampling frame, the one that comes with the R package "`sampling`" (the dataframe `swissmunicipalities`).

3. Analysis of the frame data and manipulation of auxiliary information (steps 1 and 2)

As a first step, we have to define a *frame* dataframe containing the following information:

1. a unique identifier of the unit (no restriction on the name, may be "cod");
2. the (optional) identifier of the stratum to which the unit belongs;
3. the values of m auxiliary variables (named from X1 to Xm);
4. the (optional) values of p target variables (named from Y1 to Yp);
5. the value of the domain of interest for which we want to produce estimates (named "domainvalue").

For example:

```
> head(frame)
   cod domainvalue     strato X1 X2 X3        Y1        Y2
1  100           4 4so1b4sau1  2  4  1 3283.2128 1167.9092
2  200           4 4so1a6sau1  1  6  1 1997.4587  614.9569
3  300           4 4so1a6sau1  1  6  1  569.9164 1498.6392
4  400           4 4so1a8sau1  1  8  1 1786.8751 1051.1127
5  900           4 4so1a5sau1  1  5  1  910.3036  808.0705
6 1200           4 4so1b1sau2  2  1  2 3273.3433  969.6291
```

By typing the following statement in the R environment:

```
> library(sampling)
> data(swissmunicipalities)
```

we get the "`swissmunicipalities`" dataframe, that contains 2896 observations (each observation refers to a Swiss municipality). Among the others, there are the following variables (data are referred to 2003):

REG: Swiss region.
Nom: municipality name.
Surfacesbois: wood area.
Surfacescult: area under cultivation.
Alp: mountain pasture area.
Airbat: area with buildings.
Airind: industrial area.
Pop020: number of men and women aged between 0 and 19.
Pop2040: number of men and women aged between 20 and 39.
Pop4065: number of men and women aged between 40 and 64.
Pop65P: number of men and women aged between 65 and over.
POPTOT: total population.

First, we define the identifier of the frame:

```
> swissframe$id <- swissmunicipalities$Nom
```

Let us suppose we want to plan a survey whose target estimates are the totals of population by age class in each Swiss region. In this case, our Ys variables will be:

Y1: number of men and women aged between 0 and 19.
Y2: number of men and women aged between 20 and 39.
Y3: number of men and women aged between 40 and 64.
Y4: number of men and women aged between 65 and over.

So we execute the following statements:

```
> swissframe$Y1 <- swissmunicipalities$Pop020
> swissframe$Y2 <- swissmunicipalities$Pop2040
> swissframe$Y3 <- swissmunicipalities$Pop4065
> swissframe$Y4 <- swissmunicipalities$Pop65P
```

As for the auxiliary variables (Xs), we can use all of those characterising area use (wood, mountain or pasture, cultivated, industrial, with buildings). As these variables are of the continuous type, first we have to reduce them in a categorical (ordinal) form. A suitable way to do so, we can apply a k-means clustering method (available in Rcmdr package):

```
> swissframe$X1 <- bin.var(swissmunicipalities$POPTOT, bins=18,
method='natural',labels=c('1','2','3','4','5','6','7','8','9','10','11',
'12','13','14','15','16','17','18'))
> swissframe$X2 <- bin.var(swissmunicipalities$Surfacesbois, bins=3,
  method='natural', labels=c('1','2','3'))
> swissframe$X3 <- bin.var(swissmunicipalities$Surfacescult, bins=3,
  method='natural', labels=c('1','2','3'))
> swissframe$X4 <- bin.var(swissmunicipalities$Alp, bins=3,
  method='natural', labels=c('1','2','3'))
> swissframe$X5 <- bin.var(swissmunicipalities$Airbat, bins=3,
  method='natural', labels=c('1','2','3'))
> swissframe$X6 <- bin.var(swissmunicipalities$Airind, bins=3,
  method='natural', labels=c('1','2','3'))
```

Now, we have six different auxiliary variables of the categorical type, the first with 18 different modalities, the others with 3 modalities.

Finally, we have to set the values of the "domainvalue" variable, which is mandatory. As we want to obtain estimates for each region, we set:

```
> swissframe$domainvalue <- swissmunicipalities$REG
```

Now, the "swissframe" dataframe looks like in this way:

```
> head(swissframe)
  progr REG X1 X2 X3 X4 X5 X6         id    Y1     Y2     Y3    Y4 domainvalue
1     1   4 18  3  2  1  3  3     Zurich 57324 131422 108178 66349           4
2     2   1 17  1  1  1  3  2     Geneve 32429  60074  57063 28398           1
3     3   3 17  1  1  1  3  3      Basel 28161  50349  53734 34314           3
4     4   2 17  2  3  1  3  3       Bern 19399  44263  39397 25575           2
5     5   1 17  2  2  1  3  2   Lausanne 24291  44202  35421 21000           1
6     6   4 16  3  3  1  3  3 Winterthur 18942  28958  27696 14887           4
```

that is the format required by the package.

We write the dataframe to a tab delimited file:

```
> write.table (swissframe, "swissframe.txt", row.names=FALSE, col.names=TRUE,
sep="\t", quote=FALSE)
```

In any case, this dataframe comes with the package "SamplingStrata": it can be made available by typing:

```
> data(swissframe)
```

4. Construction of atomic strata and association of the information related to target variables (steps 3 and 4)

The *strata* dataframe reports information regarding each stratum in the population. There is one row for each stratum. The total number of strata is given by the number of different combinations of X's values in the frame. For each stratum, the following information is required:

1 the identifier of the stratum (named "stratum" or "strato"), concatenation of the values of the X's variables;
2 the values of the m auxiliary variables (named from X1 to Xm) corresponding to the ones in the frame;
3 the total number of units in the population (named "N");
4 a flag indicating if the stratum is to be censused or sampled (named "cens");
5 a variable indicating the cost of interviewing per unit in the stratum (named "cost");
6 for each target variable y, its mean and standard deviation, named respectively "Mi" and "Si")
7 the value of the domain of interest to which the stratum belongs (named "DOM1" and corresponding to variable "domainvalue" in the *frame* dataframe).

For example:

```
> head(strata)
```

```
  stratum   N        M1        M2        S1        S2 cost cens DOM1 X1 X2 X3
1  1*1*1 156  623.4663  843.2696 469.92162 355.71351    1    0    1  1  1  1
2  1*1*2  68 1062.4884  867.4100 504.12793 366.40575    1    0    1  1  1  2
3  1*1*3  17  937.9182  905.4114 505.92665 327.92656    1    0    1  1  1  3
4  1*1*4  20 1377.0881  787.4087 359.69583 394.92049    1    0    2  1  1  4
5  1*1*5   3 1614.3787  660.2262  20.33451 250.12945    1    0    2  1  1  5
6  1*1*7   2 1809.0502 1324.6433 185.48919  86.84577    1    0    2  1  1  7
```

If in the *frame* dataframe are also present the values of the target y variables (from a census, or from administrative data), it is possible to automatically generate the *strata* dataframe by invoking the `buildStrataDF` function.

Let us consider again the swissframe dataframe that we have in built in previous steps. First, we have to read it from the external file that we had created:

```
> frame <- read.delim("swissframe.txt")
```

Then we can execute the function:

```
> swissstrata <- buildStrataDF(swissframe)
```

The function takes as unique argument the name of the frame, and writes out in the working directory the strata file, always named "strata.txt".

This is the structure of the created dataframe:

```
> str(swissstrata)
'data.frame':   641 obs. of  19 variables:
 $ STRATO: Factor w/ 295 levels "1*1*1*1*1*1",..: 1 2 3 5 7 8 9 10 12 14 ...
 $ N     : int  184 1 2 11 9 8 1 1 1 1 ...
 $ M1    : num  48.3 98 57 77.7 58.2 ...
 $ M2    : num  49.4 106 64 81.2 61.6 ...
 $ M3    : num  61.4 116 70 92.4 66.8 ...
 $ M4    : num  28.4 43 50 47 36.2 ...
 $ S1    : num  26.89 0 5.66 15.99 27.01 ...
 $ S2    : num  28.6 0 0 19.6 21.5 ...
 $ S3    : num  32.72 0 1.41 17.86 26.41 ...
 $ S4    : num  14.7 0 21.2 11.7 16.4 ...
 $ cost  : int  1 1 1 1 1 1 1 1 1 1 ...
 $ cens  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ DOM1  : int  1 1 1 1 1 1 1 1 1 1 ...
 $ X1    : int  1 1 1 1 1 1 1 1 1 10 ...
 $ X2    : int  1 1 1 1 2 2 2 2 3 1 ...
 $ X3    : int  1 1 1 2 1 1 1 2 1 1 ...
 $ X4    : int  1 1 2 1 1 2 3 1 2 1 ...
 $ X5    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ X6    : int  1 2 1 1 1 1 1 1 1 1 ...
```

It is worth while to note that the total number of different atomic strata is 641, lower than the dimension of the Cartesian product of the Xs (which is 4374): this is due to the fact that not all combinations of the value of the auxiliary variables are present in the sampling frame.

Variables "cost" and "cens" are initialised respectively to 1 and 0 for all strata. It is possible to give them different values:

- for variable "cost", it is possible to differentiate the cost of interviewing per unit by assigning real values;
- for variable "cens", it is possible to set it equal to 1 for all strata that are of the "take-all" type (i.e. all units in that strata must be selected).

The "swissstrata" dataframe comes together with "SamplingStrata" package, it can be made available by typing:

```
> data(swissstrata)
```

On the contrary, if there is no information in the frame regarding the target variables, it is necessary to build the *strata* dataframe starting from other sources, for instance a previous round of the same survey, or from other surveys.

In this case, we need to read sample data by executing

```
> samp <- read.delim("samplePrev.txt")
```

The only difference is that computed mean and variances of the y's are sampling estimates, whose reliability should be evaluated by carefully considering their sampling variances.

In addition to the naming constraints previously introduced, this case requires that a variable named "WEIGHT" is present in the *samp* dataframe.

Then we can execute this function in this way:

```
> strata <- buildStrataDF(samp)
```

The result is much the same than the previous case: the function creates a new dataframe, "strata", and writes out in the working directory the strata file, named "strata.txt".

Note that in all cases, for each target variable Y, mean and standard deviation are calculated excluding NAs.

5. Choice of the precision constraints for each target estimate (step 5)

The *errors* (or *CVs*) dataframe contains the accuracy constraints that are required on estimates. This means to define a maximum coefficient of variation for each variable and for each domain value. Each row of this frame is related to accuracy constraints in a particular subdomain of interest, identified by the DOM1 value.

In the case of the Swiss municipalities, we choose to define this set of constraints:

```
> data(swisserrors)
> swisserrors
   DOM  CV1  CV2  CV3  CV4 domainvalue
1 DOM1 0.08 0.12 0.08 0.12           1
2 DOM1 0.08 0.12 0.08 0.12           2
3 DOM1 0.08 0.12 0.08 0.12           3
4 DOM1 0.08 0.12 0.08 0.12           4
5 DOM1 0.08 0.12 0.08 0.12           5
6 DOM1 0.08 0.12 0.08 0.12           6
7 DOM1 0.08 0.12 0.08 0.12           7
```

This example reports accuracy constraints on variables Y1, Y2, Y3 and Y4 that are the same for all the 7 different subdomains (Swiss regions) of domain level DOM1. Of course we can differentiate the precision constraints region by region.

It is important to underline that the values of "domainvalue" are the same than those in the *frame* dataframe, and corresponds to the values of variable "DOM1" in the *strata* dataframe.

If we try to determine the total size of the sample required to satisfy these precision constraints, considering the current stratification of the frame (the 641 atomic strata), we

can do it by simply using the function bethel in the package "mauss". This function requires a different specification of the constraints dataframe, that is contained in the "errors.chk" dataframe:

```
> data(swisserrors.chk)
> swisserrors.chk
   DOM  CV1  CV2  CV3  CV4
1 DOM1 0.08 0.12 0.08 0.12
```

This is more compact way to report for all subdomains the precision constraints. Of course, it does not permit to differentiate them by subdomain. In any case, the result of the application of the Bethel algorithm is:

```
> sum(bethel(swissstrata,swisserrors.chk))
[1] 893
```

That is, with no optimization of sampling strata, the required amount of units to be selected is 893. We will see that after the optimization, this number is greatly reduced.

6. The optimisation of frame stratification (step 6)

Once the *strata* and the *constraints* dataframes have been prepared, it is possible to apply the function that optimises the stratification of the frame, that is `optimizeStrata`. This function operates on all subdomains, identifying the best solution for each one of them.
The fundamental parameters to be passed to `optimizeStrata` are:

1. `errors`: the (mandatory) dataframe containing the precision levels expressed in terms of Coefficients of Variation that estimates on target variables Y's of the survey must comply
2. `strata`: the (mandatory) dataframe containing the information related to "atomic" strata, i.e. the strata obtained by the Cartesian product of all auxiliary variables X's. Information concerns the identifiability of strata (values of X's) and variability of Y's (for each Y, mean and standard error in strata)
3. `cens`: the (optional) dataframe containing the takeall strata, those strata whose units must be selected in whatever sample. It has same structure than "strata" dataframe
4. `strcens`: flag (TRUE/FALSE) to indicate if takeall strata do exist or not. Default is FALSE
5. `initialStrata`: the initial limit on the number of strata for each solution. Default is 3000
6. `addStrataFactor`: this parameter indicates the probability that at each mutation the number of strata may increase with respect to the current value. Default is 0.01 (1
7. `minnumstr`: indicates the minimum number of units that must be allocated in each stratum. Default is 2 iter Indicated the maximum number of iterations (= generations) of the genetic algorithm. Default is 20 pops The dimension of each generations in terms of individuals. Default is 50

8. `mut_chance`: mutation chance: for each new individual, the probability to change each single chromosome, i.e. one bit of the solution vector. High values of this parameter allow a deeper exploration of the solution space, but a slower convergence, while low values permit a faster convergence, but the final solution can be distant from the optimal one. Default is 0.05
9. `elitism_rate`: this parameter indicates the rate of better solutions that must be preserved from one generation to another. Default is 0.2 (20
10. `highvalue`: parameter for genetic algorithm. Not to be changed
11. `suggestions`: optional parameter for genetic algorithm that indicates one possible solution (maybe from previous runs) that will be introduced in the initial population. Default is NULL.

In the case of the Swiss municipalities, this is a possible choice of the value of the parameters:

```
outstrata <- optimizeStrata(
      errors = swisserrors,
      strata = swissstrata,
      cens = NULL,
      strcens = FALSE,
      initialStrata = 3000,
      addStrataFactor = 0.00,
      minnumstr = 2,
      iter = 100,
      pops = 20,
      mut_chance = 0.05,
      elitism_rate = 0.2,
      highvalue = 1e+08,
      suggestions = NULL
)
```

The execution of `optimizeStrata` produces the solution of 7 different optimization problems, one for each domain (Swiss region). The number of the overall resulting optimized strata is 185, less than one third of the initial 641 atomic strata.

The convergence that has been reached for the first one domain is shown in figure 1.

The graph can be interpreted in this way:

- on the horizontal axis are reported the iterations, from 1 to 50;
- on the vertical axis is reported the size of the sample required to comply to the precision constraints;
- the red line shows the trend of the mean of the 20 solutions calculated for each iteration;
- the black line shows the best solution found till the i-th iteration.

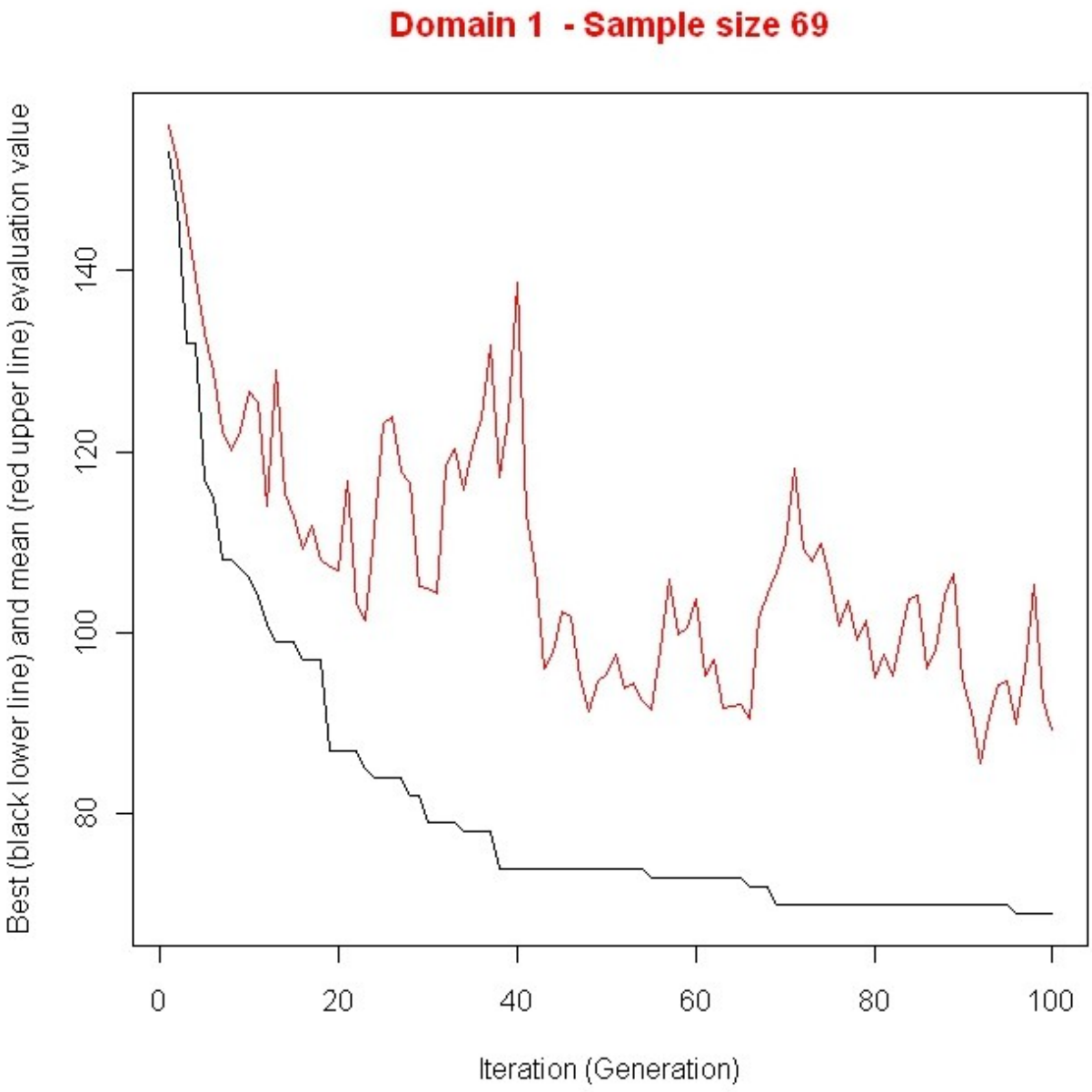


Figure 1 – Convergence of the algorithm for domain 1

The execution produces the following outputs:

1. reports on the evolution towards the best solution (one per each subdomain);
2. the file containing information on the strata corresponding to the best stratification ("outstrata.txt").

The structure of the file "outstrata.txt" is the same than the one of *strata* dataframe, plus additional information regarding:

- the identifiers of the new strata produced as aggregation of the atomic ones (*strato*);
- the allocation of units in each one of the new strata (*soluz*) .

7. Analysis of results (step 7)

Once the optimisation has been carried out, it is important to analyse the structure of the new strata. By using the function `verify` it is possible (i) to perform a graphical inspection and (ii) to verify how the allocation of units in the new strata ensures the compliance of accuracy constraints in the different domains for the whole set of target variables.

By typing

```
> data(swisserrors.chk)
> verify(swisserrors.chk,outstrata)
```

we will obtain first a graph containing the distribution of strata by numerosity in the different subdomains (figure 2).

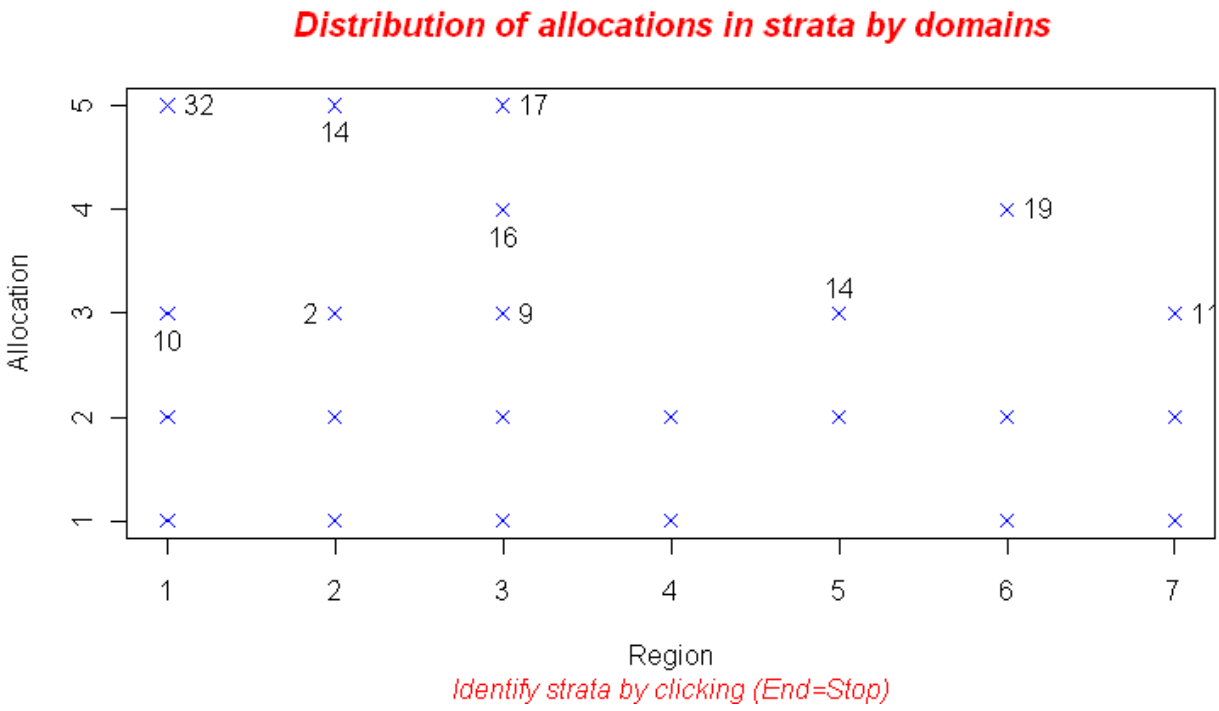
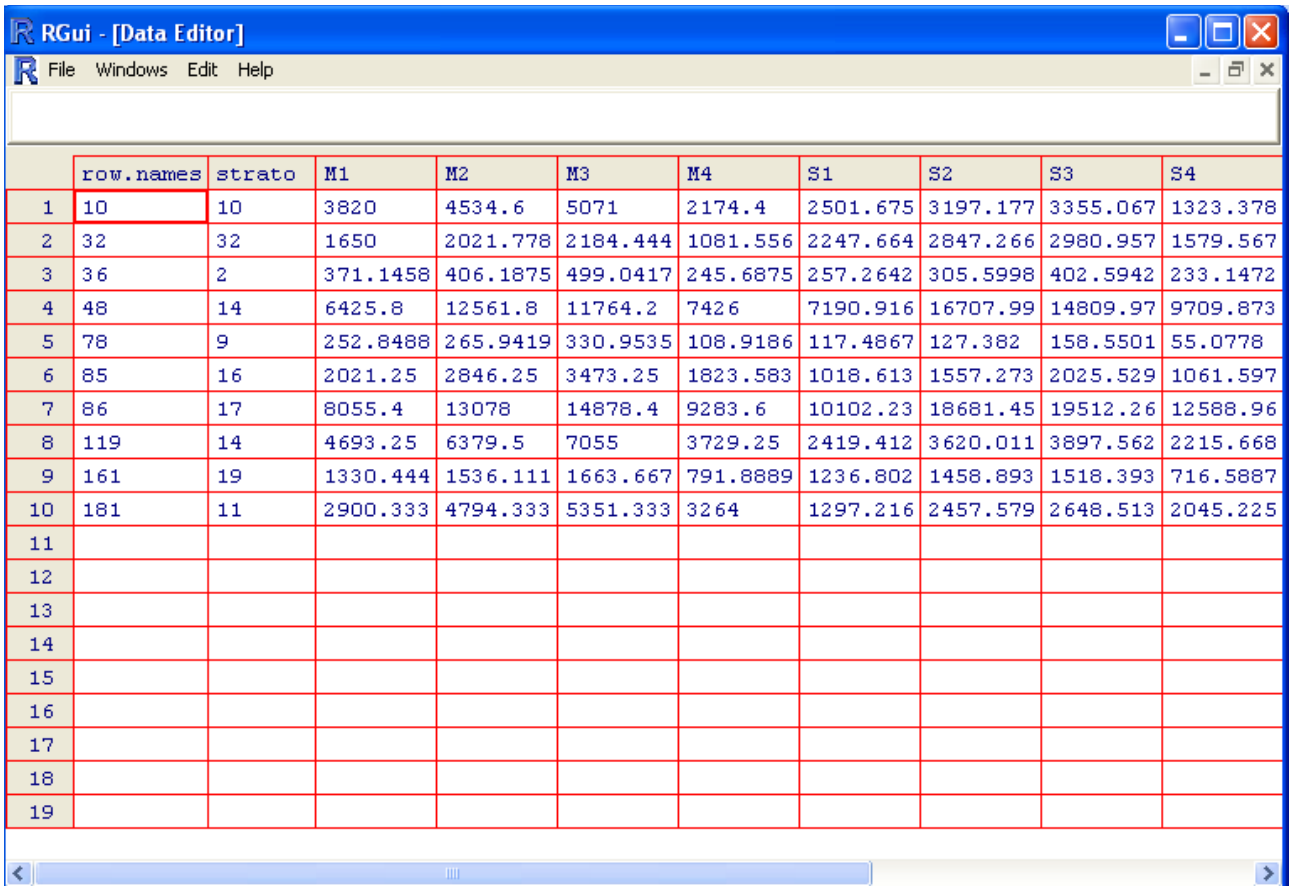


Figure 2 – Distribution of strata in different subdomains

In this graph, it is possible to click on the different points and, after stopping the identification, selected strata are presented in an edit window for the inspection.



	row.names	strato	M1	M2	M3	M4	S1	S2	S3	S4
1	10	10	3820	4534.6	5071	2174.4	2501.675	3197.177	3355.067	1323.378
2	32	32	1650	2021.778	2184.444	1081.556	2247.664	2847.266	2980.957	1579.567
3	36	2	371.1458	406.1875	499.0417	245.6875	257.2642	305.5998	402.5942	233.1472
4	48	14	6425.8	12561.8	11764.2	7426	7190.916	16707.99	14809.97	9709.873
5	78	9	252.8488	265.9419	330.9535	108.9186	117.4867	127.382	158.5501	55.0778
6	85	16	2021.25	2846.25	3473.25	1823.583	1018.613	1557.273	2025.529	1061.597
7	86	17	8055.4	13078	14878.4	9283.6	10102.23	18681.45	19512.26	12588.96
8	119	14	4693.25	6379.5	7055	3729.25	2419.412	3620.011	3897.562	2215.668
9	161	19	1330.444	1536.111	1663.667	791.8889	1236.802	1458.893	1518.393	716.5887
10	181	11	2900.333	4794.333	5351.333	3264	1297.216	2457.579	2648.513	2045.225
11										
12										
13										
14										
15										
16										
17										
18										
19										

We obtain also the following information:

```
[1] "Domain/Var.   Planned CV    Actual CV   "
[1] "1 1 / 1          0.08          0.0306"
[1] "1 1 / 2          0.12          0.0348"
[1] "1 1 / 3          0.08          0.0328"
[1] "1 1 / 4          0.12          0.038"
```

that is, the list of domain levels, domain values and variables, and related planned and expected CV, together with the "sensitivity" (the expected saving in sample size if the CV is relaxed of a 10%). This list relates only to the national level, as with the dataframe `errors.chk` we do not refer to the different domains (regions), as we do with the dataframe `errors`.

Moreover, we want also to analyse what kind of aggregation of the atomic strata the genetic algorithm did produce. To do so, we apply the function `updateStrata`, that attributes the labels of the new strata to the initial one in the dataframe *strata*, and produces:

1. a new file named 'newstrata.txt' containing all the information in the *strata* dataframe, plus the labels of the new strata;
2. a table, contained in the dataset 'strata_aggregation.txt', showing in which way the auxiliary variables X's determine the new strata.

The function is invoked in this way:

```
> newstrata <- updateStrata(swissstrata)
```

This is a part of what is contained in 'strata_aggregation.txt':

DOM1	aggr_stratum	X1	X2	X3	X4	X5	X6
1	1	2	2	2	2	1	1
1	1	3	3	2	2	1	1
1	1	4	3	1	2	1	1
1	1	5	1	2	1	1	1
1	1	7	1	1	1	1	1
1	1	7	2	1	2	1	2
1	1	11	1	1	1	2	1
1	1	11	1	2	1	2	1
1	2	5	2	2	1	2	1
1	2	9	3	3	2	2	2
1	3	3	3	1	2	1	1
1	3	6	2	2	2	1	2
1	3	14	1	3	1	2	3

For instance, the new stratum identified by the label '1' is characterised by 8 different combinations of values of the six variables X1, X2, X3, X4, X5 and X6. The new stratum '2', is given by 2 different combinations, while the new stratum '3' is individuated by 3 different combinations.

8. Updating the frame and selecting the sample (steps 8 and 9)

Once the optimal stratification has been obtained, to be operational we need to accomplish the following two steps:

1. to update the frame units with new stratum labels (combination of the new values of the auxiliary variables Xs);
2. to select the sample from the frame.

As for the first, we execute the following command:

```
> framenew <- updateFrame(frame = swissframe, strata = newstrata)
```

This function receives as arguments the indication of the dataframe in which the frame is memorised, and of the dataframe produced by the execution of the `updateStrata` function.
The execution of this function produces a dataframe "framenew", and also a file (named "framenew.txt") with the labels of the new strata produced by the optimisation step.
The allocation of units is contained in the "*soluz*" column of the dataset "outstrata.txt". At this point it is possible to select the sample from the new version of the frame:

```
> selectSample (frame = framenew, outstrata = outstrata)
```

that produces two excel files:

1. "sample.xls" containing the units of the frame that have been selected, together with the weight that has been calculated for each one of them;
2. "sample.chk.xls" containing information on the selection: for each stratum, the number of units in the population, the planned sample, the number of selected units, the sum of their weights that should equalise the number of units in the population.

The structure of "sample.chk.xls" is reported in the following:

domainvalue	strato	Nh_frame	Nh_strata	planned_units	selected_units	sum_of_wgts
1	1	13	13	3	3	13
1	2	2	2	2	2	2
1	3	3	3	2	2	3
1	4	13	13	2	2	13
1	5	5	5	2	2	5
1	6	1	1	1	1	1
1	7	22	22	2	2	22
1	8	4	4	2	2	4
1	9	69	69	2	2	69
1	10	10	10	2	2	10
1	11	12	12	2	2	12
1	12	5	5	2	2	5
1	13	8	8	2	2	8
1	14	1	1	1	1	1
1	15	5	5	2	2	5
1	16	1	1	1	1	1
1	17	7	7	2	2	7
1	18	12	12	3	3	12

It can be seen that, if the selection has been carried out correctly, the following relations must hold:

1. the number of selected units is equal to the number of planned units;
2. the sum of the weights associated to the selected units is equal to the population in the strata and in the frame.

References

Ballin M., Barcaroli G., "Optimal stratification of sampling frames in a multivariate and multidomain sample design", Contributi ISTAT n.10/2008
http://www.istat.it/dati/pubbsci/contributi/contributi2008.html