Brief overview of survey sampling techniques with R

Camelia Goga (camelia.goga@univ-fcomte.fr)
Université de Bourgogne Franche-Comté

ABSTRACT

For the last decade, many R packages have been suggested for performing sample selection according to various sampling designs as well as design-based estimation of finite population interest parameters and of their variance. The goal of this article is to give a brief overview of some of these packages with application on two datasets. A special attention is given to calibration issues.

Keywords*: over-calibration, penalized calibration, principal component analysis*

JEL Classification: *C83*, *C88*

1 Introduction

For the last decade, we assist to a large expansion of R packages dedicated to survey sampling techniques from few packages to more than eighty nowadays. A comprehensive list of all packages dedicated to survey sampling techniques and official statistics is given by Matthias Templ at https://cran.r-project.org/web/views/OfficialStatistics.html.

Broadly speaking, there are two main R-packages dedicated to survey sampling techniques: `sampling` and `survey`. Package `sampling` was suggested by Alina Matei and Yves Tillé from the University of Neuchâtel and is concerned mainly by performing sample selection according to various with or without replacement sampling designs. Some estimation issues are also treated. Package `survey` was suggested by Thomas Lumley from the University of Auckland and is concerned with design-based estimation of finite population interest parameters and of their variance. The book *Complex surveys, a guide to analysis using R* of T. Lumley (Lumley (2010)) describes most of functions contained in this package. The other existing packages are dedicated to a specific issue from survey sampling. We focus mainly in this paper on `sampling` and `survey` packages.

The paper is structured as follows: we describe in section 2 the dataset used in this paper. Section 3 gives the R-functions performing the most important sampling designs and the Horvitz-Thompson estimation of finite population parameters and of their variance. Section 4 is dedicated to the calibration method which is a very popular method used in many statistical institutes. We give first the description and the R-implementation of the calibration estimator. Next, we present the penalized and the principal components calibration estimator suggested in order to robustify the calibration estimator in presence of a very large number of auxiliary data. Finally, section 5 concludes the paper.

2 Data description

We consider in this article the dataset called `rec99` which contains an extract from the last French Census performed in 1999. Data is collected on $N = 554$ French towns from the south of France (the region Haute-Garonne) about:

- `POPSDC99`: the number of habitants in 1999
- `LOG`: the number of dwellings
- `LOGVAC`: the number of empty dwellings.

This dataset contains also the postal code (variable `CODE_N`) and the name of each small town (variable `COMMUNE`). Two factor variables are also available. The first one is called `BVQ` ("bassin de vie quotidien") and it represents a code given by INSEE, the French Statistical Institute, to each small sub-regions of the region Haute-Garone. This variable will be used as a cluster variable. The second factor variable is called `stratlog` and reflects the town population size taking four modalities: "=1" small, ..., "=4" very large. This variable will be used as a stratification variable. We give below an extract from this data-frame.

```
> rec99=read.csv("rec99.csv")
> rec99[1:3,] ###the first 3 individuals from  rec99
  CODE_N      COMMUNE BVQ_N POPSDC99 LOG LOGVAC stratlog
1  31014     ARGUENOS 31020       57  94      1        1
2  31131    CAZAUNOUS 31020       47  56      4        1
3  31348      MONCAUP 31020       26  57      2        1
```

3 Sample selection and estimation of finite population parameters

3.1 Sample selection

Let consider a finite population $U = \{1, \ldots, N\}$ and let the sample s be selected from U according to a sampling design $p(\cdot)$. Denote by $\pi_k = P(k \in s)$ the first-order inclusion probabilities supposed to be positive for all $k \in U$.

Package `sampling` implements the selection of many equal and unequal sampling designs and most of them are without replacement. The outputs of functions dedicated to sample designs are N-dimensional vectors containing zeros for the non-selected individuals and positive values for the selected individuals. In the latter situation, this positive value represents the number of times an individual was selected and it can be equal only to one if the design is without replacement.

For example, `srswor(n,N)` implements the selection of a simple random sample without replacement of size n from a population of size N. The output is in case a vector of size $N = 554$ containing $n = 70$ ones and $N - n = 484$ zeros.

```
library(sampling)
####selection of a simple random sampling without replacement of size 70
si.rec<-srswor(70,554)
```

We can obtain the selected towns with the corresponding variables by

```
data=rec99[which(si.rec==1),]
```

Function `UPsystematic(pik)` implements the systematic sampling with unequal inclusion probabilities contained in the N-dimensional vector `pik`. For example, if we desire selecting a systematic sampling design proportional to an auxiliary variable taking values x_k for $k \in U$, then the first-order inclusion probabilities are $\pi_k = nx_k / \sum_U x_k$ which can be declared in `pik`. We select from `rec99` a systematic sample of size $n = 70$ proportional to the auxiliary variable `LOG`, total number of dwellings from each small town.

```
###selection of a proportional to size systematic sample of size n=70
###total of the auxiliary variable
attach(rec99)
tLOG=sum(LOG)
tLOG
[1] 197314
###inclusion probabilities
pk=LOG/tLOG###
pik=70*pk
###selection of the sample
sys.rec=UPsystematic(pik)
data=rec[which(sys.rec==1),])
detach(rec99)
```

The usual systematic sampling is obtained in the particular case $\pi_k = n/N$, namely `pik` $= rep(n/N, N)$.

Function `UPpoisson(pik)` implements the Poisson sampling with given unequal inclusion probabilities π_k contained in `pik`. For $\pi_k = \pi$ for all $k \in U$, we obtain the Bernoulli sampling. The balanced sampling, the Brewer, Sampford, Tillé and many other unequal sampling designs are also implemented in `sampling`. We mention also the very recent package `balancedsampling` of Anton Grafstrom implementing the balanced sampling in a very fast way thanks to the `Rcpp` package. A spatially balanced sample is obtained by using the local pivot and implemented by functions `lpm1` and `lpm2` already used for the national seashore inventory in Sweden in 2015.

A stratified sample may be selected by using the function `strata` (rather slow) and cluster or multi-stage samples with functions `cluster` and `mstage` from package `sampling`.

We give below the procedure for selecting a stratified sample from our study population. The population is divided into four strata built with respect to the population town. There are four strata and the stratum number is contained in the `stratlog` variable. To select a stratified sample, one needs to give the sample sizes to select within each strata. If these sample sizes are chosen according to a proportional or X-optimal allocation, they need to be computed before the sample selection. We will use the X-optimal allocation where the auxiliary variable is `LOG`. Data needs to be ordered with respect to the stratification variable before considering the stratified sampling.

```
#####selection of a stratified sample with LOG-optimal allocation
#####computing the LOG-optimal allocation
sdlog=tapply(LOG,stratlog,sd)
n=70; N=554
nech=round(n*N*sdlog/sum(N*sdlog))
###1  2  3  4
###2  4 12  52
#####selection of the sample
rec99.order=rec99[order(rec99$stratlog),]
```

```
stsi.rec=strata(rec99.order,"stratlog",size=nech,method="srswor")
```

Information about the selected stratified sample is obtained with `getdata`:

```
###getting the stratified sample data
stsi.data=getdata(rec,stsi.rec)
```

We mention also the package `stratification` to construct strata according to the Lavallée-Hidiroglou method and package `SamplingStrata` to determine the optimal stratification of sampling frames for multipurpose sampling surveys.

Consider now the cluster sampling. Our study population is divided into $N_c = 32$ clusters called `BVQ` ("bassin de vie quotidien") and constructed according to a cut-off of the entire region Haute-Garone into disjointed regions according to some proximity criteria. The variable `BVQ_N` from `rec99` gives the cluster number. We select $n_c = 4$ clusters by simple random sampling without replacement.

```
###selection of a cluster sample
grap.rec=cluster(rec99,clustername="BVQ_N",size=4,method="srswor")
grap.rec[1:5,]
   BVQ_N ID_unit  Prob
1  31239     276 0.125
2  31239     273 0.125
3  31239     274 0.125
4  31239     275 0.125
5  31239     262 0.125
```

Information about the selected clusters is obtained with `getdata`:

```
###getting the cluster sample data
grap.data=getdata(rec99,grap.rec)
```

3.2 Estimation of finite population totals

Let Y be the interest variable and we want to estimate the finite population total of Y:

$$t_Y = \sum_{k \in U} y_k.$$

Supposing that the first inclusion probabilities π_k are positive for all $k \in U$. Then, we can construct the unbiased Horvitz-Thompson estimator of t_Y given by:

$$\hat{t}_d = \sum_{k \in s} d_k y_k,$$

where $d_k = 1/\pi_k$ are the sampling weights. Supposing that all $\pi_{kl} > 0$, the variance of the Horvitz-Thompson estimator may be estimated unbiasedly by the Horvitz-Thompson variance estimator given by

$$\hat{V}_{HT}(\hat{t}_d) = \sum_{k \in U} \sum_{l \in U} \frac{\pi_{kl} - \pi_k \pi_l}{\pi_{kl}} \frac{y_k}{\pi_k} \frac{y_l}{\pi_l}.$$

For without replacement designs of equal size, the variance may be estimated unbiasedly also by the Yates-Grundy-Sen variance estimator:

$$\hat{V}_{YGS}(\hat{t}_d) = -\frac{1}{2} \sum_{k \in U} \sum_{l \in U} \frac{\pi_{kl} - \pi_k \pi_l}{\pi_{kl}} \left(\frac{y_k}{\pi_k} - \frac{y_l}{\pi_l} \right)^2.$$

Remark that these two variance estimators may be different. Both packages `sampling` and `survey` provide functions for computing these two variance estimators. Besides, variance estimators based on replicates weights are implemented in package `survey`.

We detail now the computation of the Horvitz-Thompson estimator $\hat{t}_d$ and of its variance by using package `survey`. The creation of a design object is necessary:

```
svydesign(id, probs=NULL, strata = NULL, fpc=NULL, data = NULL, weights=NULL
```

- `id` : the label unity, always asked; ($id0$ or $id1$ **a faire tilde** means no cluster)
- `strata`: stratification variable;
- `weights` : formula or vector of inclusion probabilities of size equal to sample size;
- `fpc` : formula or vector with the finite population correction (same size as `weights`); if `fpc` not specified, then the sample is with replacement;
- `data` : the sample data;
- `weights`, `fpc`, `data` are optional.

For example, in case of simple random sampling without replacement of size $n = 70$, the weights are equal to $554/70$ and we obtain:

```
##############################
###simple random sampling
###without replacement (SI)
##############################
library(survey)
######creation of the SI design object
ech.si <- svydesign(id=~CODE_N, weights=rep(554/70,70), fpc=rep(70/554,70),
              data=rec[which(si.rec==1),])
```

The sample data `ech.si` is an object and information, in particular inclusion probabilities and variable values, contained within may be obtained with `atributes` and `summary`:

```
####getting data from the object ech.si
attributes(ech.si)
$names
[1] "cluster"    "strata"     "has.strata" "prob"
 "allprob"    "call"   "variables"
[8] "fpc"        "pps"
$class
[1] "survey.design2" "survey.design"
#####inclusion probabilities and the variable values
ech.si$allprob
ech.si$variables

summary(ech.si)
Independent Sampling design
svydesign(id = ~CODE_N, weights = rep(554/70, 70), fpc = rep(70/554,
    70), data = rec99[which(si.rec == 1), ])
Probabilities:
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1264  0.1264  0.1264  0.1264  0.1264  0.1264
Population size (PSUs): 554
Data variables:
[1] "CODE_N"   "COMMUNE"  "BVQ_N"    "POPSDC99" "LOG"      "LOGVAC"   "strat
```

The finite population total is estimated with `svytotal` and the mean with `svymean`. Output of these functions give at the same time standard-deviation estimation:

```
#####the estimation of the total and standard-deviation for the SI sample
svytotal(~LOGVAC, ech.si)
       total     SE
LOGVAC 13613 2488.4
```

We give below the finite estimation total of `LOGVAC` with stratified and cluster samplings.

```
################################
###stratified simple random sampling
###without replacement (STSI)
################################
######creation of the stsi design object
######stratum size
################################
N1=221;N2=169;N3=110;N4=54
n1=nech[1],n2=nech[2],n3=nech[3],n4=nech[4]
poids_stsi=c(rep(N1/n1,n1),rep(N2/n2,n2),rep(N3/n3,n3),rep(N4/n4,n4))
fpcst=c(rep(n1/N1,n1),rep(n2/N2,n2),rep(n3/N3,n3),rep(n4/N4,n4))
ech.stsi=svydesign(id=~CODE_N,strata=~stratlog,weights=poids_stsi,
fpc~fpcst, data=stsi.data)
#####the estimation and standard deviation for stratified sample
svytotal(~LOGVAC,ech.stsi)

############################
###simple random sampling
###without replacement
###of clusters
############################
######creation of the cluster design object
nech=nrow(grap.data)
ech.grap=svydesign(id=~BVQ_N,weights=rep(32/4,nech),
data=grap.data,fpc=rep(4/32,nech))
#####the estimation and  standard deviation for cluster sample
svytotal(~LOGVAC,ech.grap)
       total     SE
LOGVAC 10512 3058.1
```

To compare different sampling designs, simulation must be performed but this is beyond the scope of this paper.

3.3 Estimation of non-linear interest parameters

If we want to estimate more complex parameters of interest θ, then most of the times θ may be obtained as the unique solution (explicite or implicite) of a population estimation equation:

$$S(\theta) = \sum_{k \in U} S_k(\theta) = 0.$$

Coefficients of linear and logistic regression, ratio and calibration estimators may be obtained in this way. For implicit parameters, the solution is approached by numerical algorithms. The estimateur of θ is $\hat{\theta}_d$, the solution of

$$\hat{S}(\hat{\theta}_d) = \sum_{k \in s} \frac{S_k(\hat{\theta}_d)}{\pi_k} = 0,$$

and, by Taylor linearization, the variance estimator is given by (Binder (1983)):

$$\hat{V}(\hat{\theta}_d) \quad = \quad A^{-1} \cdot \hat{V}_{HT}(\hat{S}(\theta)) \cdot (A^t)^{-1},$$

where $A = \hat{S}(\hat{\theta}_d)$. Package `survey` implements this Taylor variance estimator for many non-linear estimators such as the ratio or calibration estimator (see also section 4). Design-based estimation of quantiles is treated by function `svyquantile` from `survey`. Multivariate quantile with survey data is proposed by `Gmedian`.

Packages `laeken` and `convey` are dedicated to the estimation of income inequality indicators such as Gini or Theil index.

4 Improving the Horvitz-Thompson estimator for the estimation of finite population totals

It is well-known that the Horvitz-Thompson estimator may be improved by using auxiliary information and the calibration, as suggested by Deville and Särndal (1992), is one of the most popular method used in many statistical institutes.

Consider that p auxiliary variables $\mathcal{X}_1, \ldots, \mathcal{X}_p$ are available and let $\mathbf{x}_k = (X_{1k}, \ldots, X_{pk})'$ pour $k \in U$. We suppose that the total of auxiliary variables are known, namely $t_{\mathbf{x}} = \sum_{k \in U} \mathbf{x}_k$ is known.

The calibration method consists in finding new weights $\mathbf{w}_s^{\text{cal}} = (w_{ks}^{\text{cal}})_{k \in s}$ such that they are as close as possible in the sense of a distance Υ_s to the sampling designs $\mathbf{d} = (d_k)_{k \in s}$:

$$\mathbf{w}_s^{\text{cal}} \quad = \quad \text{argmin}_{\mathbf{w}} \Upsilon_s(\mathbf{w}, \mathbf{d})$$

and subject to the calibration constraints

$$\sum_{k \in s} w_{ks}^{\text{cal}} \mathbf{x}_k \quad = \quad t_{\mathbf{x}}.$$

The calibrated estimator is $\hat{t}_y^{\text{cal}} = \sum_{k \in s} w_{ks}^{\text{cal}} y_k$. Several distance functions $\Upsilon_s(\mathbf{w}, \mathbf{d})$ have been considered in Deville and Särndal (1992) such as the chi-squared, raking, logit distances and they proved that the calibration estimator is asymptotically equivalent to the estimator obtained for the chi-squared distance, $\Upsilon_s(\mathbf{w}, \mathbf{d}) = \sum_{k \in s} (w_k - d_k)^2 / d_k$. For the chi-squared distance, the calibration weights are given by

$$w_{ks}^{\text{cal}} = d_k - d_k \mathbf{x}_k' \left(\sum_{k \in s} d_k \mathbf{x}_k \mathbf{x}_k \right)^{-1} (\hat{t}_{\mathbf{x}d} - t_{\mathbf{x}}), \quad k \in s,$$

where $\hat{t}_{\mathbf{x}d} = \sum_{k\in s} d_k \mathbf{x}_k$. The calibration estimator is

$$\hat{t}_y^{\text{cal}} = \sum_{k\in s} w_{ks}^{\text{cal}} y_k = \sum_{k\in s} d_k y_k - (\hat{t}_{\mathbf{x}d} - t_{\mathbf{x}})' \hat{\boldsymbol{\beta}}_X,$$

where $\hat{\boldsymbol{\beta}}_X = (\sum_{k\in s} d_k \mathbf{x}_k \mathbf{x}_k')^{-1} \sum_{k\in s} d_k \mathbf{x}_k y_k$. The variance estimator of $\hat{t}_y^{\text{cal}}$ is given by

$$\hat{V}(\hat{t}_y^{\text{cal}}) = \sum_{k\in s}\sum_{k\in s} \frac{\pi_{kl} - \pi_k \pi_l}{\pi_{kl}} \frac{y_k - \mathbf{x}_k' \hat{\boldsymbol{\beta}}_X}{\pi_k} \frac{y_l - \mathbf{x}_l' \hat{\boldsymbol{\beta}}_X}{\pi_l}.$$

In R, several packages are concerned with calibration. Functions `calib, gencalib` in package `sampling` compute the calibration weights and the g-weights. Packages `laeken` (`calibWeights` function) and `simPop` (`calibSample` function) suggest faster implementations (depending on the example) of parts of `calib`. Function `calibrate` from package `survey` compute the calibration weights. Weights for computing the ratio estimator and respectively, the poststratified estimator are obtained with functions `svyratio` and `poststratified`. We mention also the very recent package `icarus`.

We focus now on the package `survey`. The calibration weights are obtained with the `calibrate` function:

```
calibrate(design, formula, population,variance=NULL,
      bounds=c(-Inf,Inf), calfun=c("linear","raking","logit"),...)
```

- `design`: the survey design objet;
- `formula`: model formula for calibration model
- `population`: vectors of population totals
- `calfun`: distance functions used in calibration
- `bounds`: bounds for the calibration weights
- `variance`: if not NULL, then the calibration variance is proportional to the linear combination of those columns of the model matrix

We use this function in order to estimate the population total of `LOGVAC` by taking into account the auxiliary information given by the intercept and the variable `LOG` whose total is 197314:

```
ech.cal<-calibrate(ech.si,~LOG,c(554,197314))
```

The value of this function is an object, we can extract the calibration weights with `1/ech.cal$prob` and the values of the interest variable with `ech.cal$variables`. The calibration estimator for the estimation of the total of `LOGVAC` and its estimated standard-deviation (obtained by Taylor linearization) is obtained as follows:

```
svytotal(~LOGVAC, ech.cal)
        total     SE
LOGVAC 9853.5 858.84
```

The ratio estimator may be obtained with the function `svyratio`. Since the ratio estimator is a particular case of the calibration estimator, it can be also obtained with `calibrate` by excluding the intercept and variance proportional to the single auxiliary variable (here the variable `LOG`) with the option `variance=1` :

```
ech.cal2<-calibrate(ech.si,~LOG-1,197314, variance=1)
svytotal(~LOGVAC,ech.cal2)
        total    SE
LOGVAC 9461.9 882.2
```

The very recent package `Icarus`, created by Antoine Rebecq (ex Insee, now Ubisoft, Montreal), is dedicated only to calibration method. It was implemented for users from the INSEE and this is why, it is inspired from the very popular macro SAS CALMAR ("calage sur marges"). Nonresponse can be handled and also penalized calibration, but the variance estimation is not provided, like CALMAR. The estimated total or mean with the so obtained calibration weights may be obtained with functions `weightedTotal` or `weightedMean`. A detailed description (in French) of this package is given in Rebecq (2016).

The `calibration` function is:

```
calibration(data, marginMatrix, colWeights, method = "linear", bounds = NULL, ...)
```

- `data`: survey data;
- `marginMatrix`: population totals of auxiliary variables;
- `colWeights`: the sampling weights;
- `method`: the method used to calibrate ("linear", "raking", "logit", "truncated");
- `bounds`: vector of lower and upper bounds for the calibration weights;

We use this function to estimate again the population total of `LOGVAC` by taking into account the auxiliary information given by the intercept and the variable `LOG`.

```
#####population totals
m_1=c("taillepop",0,554)
m_LOG=c("LOG",0,197314)
marges_rec=rbind(m_1,m_LOG)
########################
poids=rep(554/70,70)
taillepop=rep(1,70)

#####???of the intercept
essai=cbind(rec99[which(si.rec==1), ], taillepop, poids)

######calibration weights
poids_cal=calibration(data=essai,marginMatrix=marges_rec,colWeights="poids",
bounds=c(0.4, 2.1), description=TRUE)
#####calibration estimation of the total of LOGVAC
logvac_cal=weightedTotal(essai$LOGVAC,poids_cal)
#[1] 9853.461
```

We remark that we obtain the same estimation as with the `calibrate` function from the `survey` package.

4.1 Estimation with over-calibration

Nowadays, we are at the age of big-data and large datasets may be recorded thanks to connected smart objects such as smartphones, computers and smart meters. In this situation, a very large number of auxiliary variables may be obtained.

Performing calibration with a very large number of auxiliary information, called also over-calibration by Guggemos and Tillé (2010), can results in negative, very large and unstable calibration weights w_{ks}. Moreover, in this situation, the predefined benchmarks on the weight ratio w_{ks}/d_k may not be satisfied if for example, the chi-square distance is used. With many auxiliary variables, collinearity problems may arise and the matrix $\sum_{k\in s} d_k \mathbf{x}_k \mathbf{x}_k^T$ may not be invertible and so, the calibration weights can not be computed. And lastly, it can be remarked that the variance of the calibration estimator increases (Silva and Skinner (1997), Cardot et al. (2017), Chauvet and Goga (2017)).

Several solutions have been suggested to overcome these drawbacks. We may choose the most important variables but with multipurpose surveys, this choice may be difficult to put into practice. We may use instead all the variables and perform penalized calibration via the ridge regression (Bardsley and Chambers (1984); Chambers (1996); Rao and Singh (2009); Guggemos and Tillé (2010); Beaumont and Bocci (2008)). Very recently, Cardot et al. (2017) suggested compressing the information contained in the auxiliary variables by performing principal component calibration.

4.2 Penalized calibration

The calibration equations are "released", namely they are only approximatively satisfied and the error between $\hat{t}_{\mathbf{x}w}$ and $t_{\mathbf{x}}$ is controlled by means of a penalty. More exactly, we search for the weights $\mathbf{w}_s^{\text{pen}}(\lambda) = (w_{ks}^{\text{pen}}(\lambda))_{k\in s}$ satisfying the following penalized optimization problem:

$$\mathbf{w}_s^{\text{pen}}(\lambda) \quad = \quad \text{argmin}_{\mathbf{w}} \sum_{k\in s} \frac{(w_k - d_k)^2}{d_k} + \lambda \sum_{j=1}^{p} C_j (\hat{t}_{w,X_j} - t_{X_j})^2,$$

where $C_j, j = 1, \ldots, p$, is a user-specified cost associated with not satisfying the jth calibration equation. If $\lambda C_j = 0$, then the constrain upon t_{X_j} is not considered and if λC_j is very large, then the constraint upon t_{X_j} is exactly satisfied.

The penalized weights are given by:

$$w_{ks}^{\text{pen}}(\lambda) = d_k - d_k \mathbf{x}_k' \left(\sum_{k\in s} d_k \mathbf{x}_k \mathbf{x}_k + \lambda^{-1} \mathbf{C}^{-1} \right)^{-1} (\hat{t}_{\mathbf{x}d} - t_{\mathbf{x}}), \quad k \in s$$

and the penalized calibration estimator is a GREG-type estimator with a ridge coefficient of regression:

$$\hat{t}_{yw}^{\text{pen}}(\lambda) \quad = \quad \sum_{k\in s} w_{ks}^{\text{pen}}(\lambda) y_k = \sum_{k\in s} d_k y_k - \left(\sum_{k\in s} d_k \mathbf{x}_k - \sum_{k\in U} \mathbf{x}_k \right)' \hat{\boldsymbol{\beta}}_X(\lambda),$$

where $\hat{\boldsymbol{\beta}}_X(\lambda) = \left(\sum_{k\in s} d_k \mathbf{x}_k \mathbf{x}_k' + \lambda^{-1} \mathbf{C}^{-1} \right)^{-1} \sum_{k\in s} d_k \mathbf{x}_k y_k$. The variance estimator of $\hat{t}_{yw}^{\text{pen}}(\lambda)$ is

$$\hat{V}(\hat{t}_{yw}^{\text{pen}}(\lambda)) = \sum_{k\in s} \sum_{k\in s} \frac{\pi_{kl} - \pi_k \pi_l}{\pi_{kl}} \frac{y_k - \mathbf{x}_k' \hat{\boldsymbol{\beta}}_X(\lambda)}{\pi_k} \frac{y_l - \mathbf{x}_l' \hat{\boldsymbol{\beta}}_X(\lambda)}{\pi_l}.$$

The method has already been used at Statistics Canada and at INSEE. We can perform penalized calibration by using the package `icarus` and function `calibration`. We give below again this function with the options related to the costs C_j and the tuning parameter λ needed for performing penalized calibration:

```
calibration(data, marginMatrix, colWeights, method = "linear",
  bounds = NULL, q = NULL, costs = NULL, gap = NULL,  lambda = NULL, ...)
```

- `costs`: vector of C_j costs, they must be given;
- `gap`: $\max(w_k/d_k) - \min(w_k/d_k)$
- `lambda`: the tuning parameter λ used in penalized calibration; by default, chosen automatically by the bisection algorithm.

No need to consider bounded distances this is why only the chi-square and the raking distance are considered. Different costs may be used, a very large C_j meaning that the constraint must be exactly satisfied.

4.3 Calibration on principal components

The alternative to penalized calibration is to "compress" the information contained in the X-matrix by considering principal components analysis. Consider the principal components $\mathbf{Z}_1, \ldots, \mathbf{Z}_p$

$$\mathbf{Z}_j = \mathbf{X}\mathbf{v}_j, \quad j = 1, \ldots, p.$$

where $\mathbf{v}_j$ is the jth eigenvector associated to the jth eigenvalue λ_j of $N^{-1}\,\mathbf{X}'\mathbf{X}$. We search for weights $\mathbf{w}_s^{\mathrm{pc}} = (w_{ks}^{\mathrm{pc}})_{k\in s}$ satisfying:

$$\mathbf{w}_s^{\mathrm{pc}} = \mathrm{argmin}_{\mathbf{w}} \sum_{k\in s} \frac{(w_k - d_k)^2}{d_k}$$

subject to

$$\hat{t}_{w,\mathbf{Z}_j} = t_{\mathbf{Z}_j} \text{ for } j = 1, \ldots, r \quad \text{with} \quad r << p,$$

where $t_{\mathbf{Z}_j}$ is the total of the jth principal component $\mathbf{Z}_j$. The principal component calibration estimator is a GREG-type estimator with a principal component coefficient of regression:

$$\hat{t}_{yw}^{\mathrm{pc}}(r) \;=\; \sum_{k\in s} d_k y_k - \left(\sum_{k\in s} d_k \mathbf{x}_k - \sum_{k\in U} \mathbf{x}_k\right)' \hat{\boldsymbol{\beta}}_X^{\mathrm{pc}}(r),$$

where $\hat{\boldsymbol{\beta}}_X^{\mathrm{pc}}(r) = (\mathbf{v}_1, \ldots, \mathbf{v}_r)' \left(\sum_{k\in s} d_k \mathbf{x}_k \mathbf{x}_k'\right)^{-1} \sum_{k\in s} d_k \mathbf{x}_k y_k$. The variance estimator of $\hat{t}_{yw}^{\mathrm{pc}}(r)$ is

$$AV(\hat{t}_{yw}^{\mathrm{pc}}(r)) = \sum_{k\in s}\sum_{k\in s} \frac{\pi_{kl} - \pi_k\pi_l}{\pi_{kl}} \frac{y_k - \mathbf{x}_k'\hat{\boldsymbol{\beta}}_X^{\mathrm{pc}}(r)}{\pi_k} \frac{y_l - \mathbf{x}_l'\hat{\boldsymbol{\beta}}_X^{\mathrm{pc}}(r)}{\pi_l}.$$

If no complete auxiliary information is available, then the PC variables $\mathbf{Z}_j$ may be estimated by $\hat{\mathbf{Z}}_j = \mathbf{X}\hat{\mathbf{v}}_j$ and consider calibration on the zero-totals of $\hat{\mathbf{Z}}_j$ (Cardot, Goga and Shehzad, 2017).

With R, function `svyprcomp` of package `survey` performs principal components analysis with survey data and allows obtaining the principal component estimates when data is not complete:

`svyprcomp(formula, design, center = TRUE, scale. = FALSE, scores = FALSE, ...)`

- `formula`: model formula describing variables to be used
- `design`: the survey design object
- `center = TRUE`: data is centered by default
- `scale. = FALSE`: data is not scaled by default
- `scores = FALSE`: scores on each component needed for biplot are not returned.

The value is an object similar to `prcomp` but with survey design information. Using `svyprcomp`, we can compute the estimated $\hat{\mathbf{Z}}_j$ which can be used next as calibration variables in `calibrate` or `calibration`.

5 Conclusion

We presented a brief overview of the most used packages dedicated to survey sampling techniques with implementation on a real dataset. Nonresponse is often in survey sampling and many packages on missing data exist but most of them do not taken into account sampling weights. There are still many fields in survey sampling theory which are not implemented in R such as nonparametric estimation with survey data, robust estimation for totals, indirect sampling.

REFERENCES

1. **Bardsley, P. and Chambers, R.**, 1984, *Multipurpose estimation from unbalanced samples*. Applied Statistics, 33:290–299.
2. **Beaumont, J.-F.** and **Bocci, C.**, 2008, *Another look at ridge calibration*. Metron-International Journal of Statistics, LXVI:260–262.
3. **Binder, D. A.**, 1983, *On the variances of asymptotically normal estimators from complex surveys*. International Statistical Review, 51:279–292.
4. **Cardot, H., Goga, C.**, and **Shehzad, M.-A.**, 2017, *Calibration and partial calibration on principal components when the number of auxiliary variables is large*. Statistica Sinica, 27(243-260).
5. **Chambers, R.**, 1996, *Robust case-weighting for multipurpose establishment surveys*. Journal of Official Statistics, 12:3–32.
6. **Chauvet, G**. and **Goga, C.**, 2017, *Bootstrap variance estimation and variable selection*. in work.
7. **Chauvet, G.** and **Tillé, Y.**, 2006, *A fast algorithm of balanced sampling*. Computational Statistics, 21:53–61.
8. **Deville, J.-C.** and **Särndal, C.-E.**, 1992, *Calibration estimators in survey sampling*. Journal of the American Statistical Association, 87:376–382.12
9. **Guggemos, F.** and **Tillé, Y.,** 2010, *Penalized calibration in survey sampling: Designbased estimation assisted by mixed models*. J. of Statistical Planning and Inference, 140:3199–3212.
10. **Lumley, T.**, 2010, *Complex surveys, a guide to analysis using R*. Wiley Series in Surveys Methodology.
11. **Rao, J.** and **Singh, A. C**., 2009, *Range restricted weight calibration for survey data using ridge regression*. Pakistan Journal of Statistics, 25(4):371–384.
12. **Rebecq, A.**, 2016, *Icarus : un package r pour le calage sur marges et ses variantes*. In 9eme Colloque francophone sur les sondages, 2016 Gatineau, Canada.
13. **Silva, P.** and **Skinner, C.**, 1997,. *Variable selection for regression estimation in finite populations*. Survey Methodology, 23:23–32.