Enchancing the package REVIVALS

GUILLEM FORTÓ CORNELLA

Master thesis

Toulouse School of Economics Master 2, Statistics and Econometrics

2019 / 2020

June 2020 - September 2020

Key words: `REVIVALS`; Conditional bias; Outliers; Design-based inference; Robust estimator

Contents

List of Figures

List of Tables

General context

Given the situation provoked by the Covid-19 pandemic, I found myself obliged to cancel an internship that had to take place in Paris during summer 2020. I therefore started to think about a thesis subject, and remembered that I specially appreciated the Survey Sampling course taught by Anne Ruiz-Gazen during the first semester of my Master 2. After discussing about it, we agreed that I could work on the theme of robust estimation through conditional bias methods. Cyril Favre-Martinoz had already started to build an R package to make some of the already published theoretical results more accessible, but there was still some work to be done. We agreed on virtually meeting once a week in order for me to update Anne and Cyril on my progress.

Specifically, my mission was not only to enhance the package by making it more robust and user-friendly, but also to illustrate all the new features in an article that will likely be published. I have been teleworking with my personal computer, so no need to mention commuting times nor any new human relationship like in an ordinary internship. My interest in the subject, the meeting discussions, as well as the possibilities of application to real data, have been my main source of motivation.

Abstract

The presence of influential values in the estimation of population totals is an important problem that numerous polling organizations have to commonly deal with. While winsorisation methods seem to have become the most common technique to treat them, Beaumont et al. (2013) proposed to use the conditional bias to construct an equivalent robust estimator that lessens the effect of these outliers. C. Favre-Martinoz started to build the package `REVIVALS` with the idea of making these theoretical results available to the general R user. Resuming the work that he had been doing so far, we present several improvements to the package. These include some robustness enhancements in line with the theoretical conditions exposed in the literature, as well as some new features that facilitate its use. We also propose a real example application with some data about vacant homes collected in the Haute-Garonne French department. This illustrates the functions of `REVIVALS` and gives some lines of thought about the behaviour of the Horvitz-Thompson robust estimator in the presence of influential values.

Acknowledgements

I would like to express my gratitude to my teacher and project supervisor Dr. Anne Ruiz-Gazen for putting great effort into providing guidance whenever needed. I am also very grateful to Dr. Cyril Favre Martinoz, with whom I've worked all along the drafting of the article.

Finally, I would like to thank my family for their invaluable support.

Abbreviation / Acronyms

- HT: Horvitz-Thompson
- RHT: Robust Horvitz-Thompson
- SI / SRSWOR: Simple Random Sampling Without Replacement
- CB: Conditional Bias
- DT: Dalén-Tambay
- BHR: Beaumont Haziza Ruiz-Gazen
- ST: Stratified
- INSEE: Institut National de la Statistique et des Etudes Economiques

1 Introduction

The treatment of influential units is key in situations where the variables collected in a survey have a strongly asymmetrical distribution. In business surveys, for example, equally sized companies may have very different incomes, and so the density of the revenues variable can happen to be highly skewed. These data points, which differ significantly from the main bulk of data, is what we call *outliers*.
In survey sampling, errors are corrected at what is known as the editing stage, to the point where it is assumed that there are no errors left once we reach the estimation step. That is to say that all the outliers left are supposed to come from atypical observations, and not from any typing error nor any other kind of measurement mistake. As the goal is to make some inference on the entire population, this means that we don't want to discard them.
Of course, an outlier can sometimes cause serious trouble in the estimation step. Their presence may lead to unstable estimators which distort the final result. And despite what one may think, the use of robust estimators from classical statistics is not recommended in these kind of situations. These may indeed drastically downweight some valid data and lead to highly biased estimators. Instead, we typically try to use robust estimators, capable of detecting 'influential values' and which serve as protection against extreme observations.
The notion of 'influential value' needs to be adapted to the sampling context. Beaumont et al. (2013) proposes to use the conditional bias as a measure of influence, which we will further explain in a following section.

Given that we are able to measure this influence for a given estimator, and that we know some robust estimators to make inference in a design-based approach, how do we build a good R package that makes the estimations more accessible?

After this first introduction, in the second part of this report we start by giving an overview of the state of the theory of robustness estimation in finite population. Here we will only focus on the design approach, which means that randomness is only present at the sampling stage and no assumptions are made on the data distribution.

The third part is dedicated to the explanation of the REVIVALS R package. We first detail the purpose of each function, trying to make the link between theory and practice as much as possible.

In what follows are listed the main contributions that have been proposed in the scope of this thesis, in order to improve the robustness and the ease of use of the package.

Finally, we present an example of application using some real data with the wrapper function of the package.

2 Literature review

2.1 Sampling designs

Once the statistician has identified the target population and defined the survey frame, one of the next main steps of a survey is the determination of the sampling design. The most common type of sampling is random sampling, where the probabilities according to which the units are drawn are known. Here we will present the designs implemented in `REVIVALS`, which we will denote as follows:

- srswor / si: Simple Random Sampling Without Replacement
- poisson: Poisson Sampling
- rejective: Rejective Sampling

Let U be the finite population of N units: $U = \{1, \dots, N\}$.

Let $y_i = (y_{i1}, \dots, y_{ip})^T$ be a vector of p variables of interest associated to unit $i \in U$. The variables of interest are the ones we attempt to measure in the survey.

A sample is usually defined by a vector of sample indicators $s = (I_1, \dots, I_i, \dots, I_N)^T$, where

$$I_i = \begin{cases} 1 & \text{if } i \in s \\ 0 & \text{otherwise.} \end{cases}$$

Then the sample size is simply expressed as: $n = \sum_{i \in U} I_i = card(s)$.

We also need to define the concept of first-order and second-order inclusion probabilities:

$$\pi_i = \mathbb{P}(i \in s) \quad \text{and} \quad \pi_{ij} = \mathbb{P}(i \in s,\ j \in s)$$

respectively, with $\pi_{ii} = \pi_i$.

Finally, the design weight of unit i is defined as the inverse of its inclusion probability: $d_i = \frac{1}{\pi_i}$.

The next page presents two tables summarizing our sampling designs.

Design	**Description**	π_i	π_{ij}
SI / SRSWOR	**We draw a sample of size n with equiprobability among all possible samples**	$\dfrac{n}{N}$	$\dfrac{n(n-1)}{N(N-1)}, \forall i, j \in U$
Poisson / Bernoulli	**We fix $\pi = (\pi_1, \pi_2, ..., \pi_N)$ and randomly draw each observation independently. If all π_i are equal, then it is called a Bernoulli design. The sample size is random.**	**(fixed)**	$\pi_i\pi_j$**,** $i \neq j$ / π^2**,** $i \neq j$
Rejective	**We fix $\pi = (\pi_1, \pi_2, ..., \pi_N)$ and follow a Poisson design, rejecting all samples until we obtain one of size n.**	**(fixed)**	$\simeq \pi_i\pi_j(1-(1-\pi_i)(1-\pi_j)D^{-1} + o(D^{-2})$ **for** $D = \sum_{i=1}^{N} \pi_i(1-\pi_i)$ **large**

Table 1: Non-stratified sampling designs used in `REVIVALS`

Each of the designs in Table 1 can also be implemented in its stratified version. It consists in dividing the population U into H strata according to an auxiliary variable (known for all the population), and drawing a sample s_h of size n_h among the N_h units of every stratum independently. For example, if we chose STSRSWOR (STratified Simple Random Sampling Without Replacement), we would have:

Design	**Description**	π_i	π_{ij}
STSI / STSRSWOR	/	$\dfrac{n_h}{N_h}$	$\begin{cases} \frac{n_h n_k}{N_h N_k} & \textbf{, if } i \in U_h, j \in U_k, h \neq k \\ \frac{n_h(n_h-1)}{N_h(N_h-1)} & \textbf{, if } i, j \in U_h \end{cases}$

Table 2: STSRSWOR summary

Considering Stratified Poisson sampling (STPoisson) and Stratified Rejective sampling (STRejective), this yields a total of 6 possible sampling designs.

2.2 Conditional bias

In a finite population framework, outliers cannot simply be excluded from the group of observations of a survey sample, because we consider that an editing phase has already been done beforehand. In other words, we have already made sure that there are no missing or inconsistent values (e.g. an observation given in euros instead of thousands of euros), nor any invalid input errors left, and so we cannot remove them because they are part of our population. This means that they must also intervene in the interest parameter computation that we are seeking to estimate, just as any other non-outlier observation.

Say we have a population of technological companies that we are surveying because we want to estimate their total sales revenue. If among these companies there are big businesses like Apple or Amazon, we know they are not to ignore because they would distort the result otherwise. Hence, instead of discarding them, what is done in practice is rather the opposite: ensuring we survey them by using a stratified design and putting them all in a stratum. If we set their probability of being drawn to 1, that is, we do a census on these observations, then the total in their stratum is known with certainty. This acts as a prevention from the risk of not drawing them under a non-stratified sampling design.

It therefore seems important to find a way to measure the influence of a given unit i in the estimation of a total.

It turns out that the conditional bias, introduced by Moreno-Rebollo et al. (1999). [MMM99], is a good measure of the contribution of a unit i to the variance of the total estimator. The idea is to say: if we do not put any condition about unit i being contained in the sample, how different would the estimation error be from the one obtained by making sure that i is in the sample (i.e. $I_i = 1$).

Note that in survey sampling, to consider all possible samples is mathematically equivalent to using a design based expectation operator in our expression. That is to say, if we had a population of $N = 3$ units, then computing the expected error of a parameter of interest θ would come down to compute $\hat{\theta} - \theta$ for each of the following samples of observations: $\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$. These are all the $2^N = 8$ possible subsets of $U = \{1, 2, 3\}$.

Hence, the first scenario, with no conditioning, simply corresponds to the bias of the estimator:

$$Bias(\hat{\theta}) = \mathbb{E}(\hat{\theta}) - \theta = \mathbb{E}(\hat{\theta} - \theta),$$

whereas the second one is

$$B_i^{\hat{\theta}}(I_i = 1) = \mathbb{E}(\hat{\theta} - \theta \mid I_i = 1). \tag{1}$$

This second expression (1) is what is defined as the conditional bias (CB) of a unit i.

If we consider the Horvitz-Thompson (HT) estimator $\hat{\theta} = \sum_{i \in s} d_i y_i$, the conditional bias of a sample unit i is defined as (cf. [BHR13]):

$$B_i^{\mathrm{HT}}(I_i = 1) = \sum_{j \in U} \left(\frac{\pi_{ij}}{\pi_i \pi_j} - 1 \right) y_j, \tag{2}$$

A good property of the CB is that a unit i drawn with certainty in the sample (i.e. $\pi_i = 1$) yields a zero-influence measure:

$$B_i^{\mathrm{HT}}(I_i = 1) = \sum_{j \in U} \left(\frac{\pi_j}{1 \times \pi_j} - 1 \right) y_j = 0$$

For the sake of clarity, $B_i^{\mathrm{HT}}(I_i = 1)$ will simply be denoted B_i^{HT} and the estimator of the conditional bias B_i^{HT} will be denoted $\hat{B}_i^{\mathrm{HT}}$. Here is its expression for four of the different designs implemented in `REVIVALS`:

- **SRSWOR**

  We have: $B_i^{\mathrm{HT}} = \frac{N}{N-1} \left(\frac{N}{n} - 1 \right) (y_i - \bar{Y}_U)$, for $i \in U$, where $\bar{Y}_U = \sum_{i \in U} \frac{1}{N} y_i$.

  This CB is unknown and so we estimate it by:

  $$\forall i \in U,\ \hat{B}_i^{\mathrm{HT}} = \frac{n}{n-1} \left(\frac{N}{n} - 1 \right) (y_i - \bar{y}), \tag{3}$$

  where $\bar{y} = \frac{1}{n} \sum_{i \in S} y_i$. We can also replace $\bar{y}$ by the median in the sample. The conditions to check are that $n > 1$ and N is known.

- **STSRSWOR**

  We have: $B_i^{\mathrm{HT}} = \frac{N_h}{N_h-1}\left(\frac{N_h}{n_h}-1\right)(y_i - \bar{Y}_{U_h}),$ for $i \in U_h$, where $\bar{Y}_{U_h} = \sum_{i\in U_h}\frac{1}{N_h}y_i$.

  This CB is unknown and so we estimate it by:

$$\hat{B}_i^{\mathrm{HT}} = \frac{n_h}{n_h-1}\left(\frac{N_h}{n_h}-1\right)(y_i - \bar{y}_h),\ \text{for}\ \ i \in U_h, \tag{4}$$

  where $\bar{y}_h = \sum_{i\in s_h} y_i$. We can also replace $\bar{y}_h$ by the median in the stratum.

  The conditions to check are that $n_h > 1$, N_h are known and all $N_h \neq 0$.

- **Poisson**

  We have:

$$B_i^{\mathrm{HT}} = \left(\frac{1}{\pi_i}-1\right)\ y_i. \tag{5}$$

  In this case there is no need to estimate the conditional bias.

- **Rejective**

  We have: $B_i^{\mathrm{HT}} = (\frac{1}{\pi_i}-1)(y_i - B\pi_i)$ where $B = D^{-1}\sum_{j\in U}(1-\pi_j)y_j$ and $D = \sum_{i\in U}\pi_i(1-\pi_i)$.

  This CB is unknown and so we estimate it by:

$$\Rightarrow \hat{B}_i^{\mathrm{HT}} = \frac{\sum_{j\in s}(\frac{1}{\pi_j}-1)y_j}{\sum_{j\in s}(1-\pi_j)}. \tag{6}$$

  The conditions to check are that D is large enough and $N/D = O(1)$ (i.e. bounded by 1).

As we can see, the conditional bias strongly depends on the survey design, and seems 'easily' extendable to new designs. It is also possible to write it for the stratified Poisson and the stratified Rejective designs.

Application

To see an application example, we have at our disposal the *rec99htegne* data set, which contains $N = 554$ observations and $p = 7$ variables regarding the French communes in the Haute-Garonne department, from a 1999 census. Here is its header:

```
          CODE_N   COMMUNE           BVQ_N    POPSDC99LOG       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
4         31447    RAZECUEILLE       31020    37       89       6        1
5         31140    CHEIN-DESSUS      31020    184      174      28       2
```

LOGVAC, which stands for the French ‘logements vacants’, corresponds to the number of vacant homes in each commune. It will be our main interest variable, and the one for which we will try to estimate the total later on. For now, let us just see how does the conditional bias vary for a given sampling design.

First of all, here are some basic descriptive statistics for LOGVAC:

Min	Q1	Median	Mean	Q3	Max	$\mathbb{V}ar$	Kurtosis	Skewness
0.00	4.00	8.00	19.44	20.00	350.00	1 104.50	29.450	4.533

Table 3: LOGVAC descriptive statistics

Its density plot looks like this:

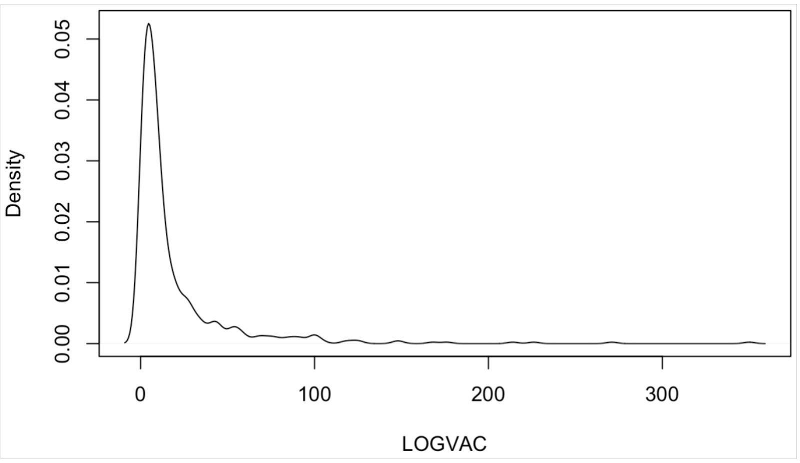


Figure 1: LOGVAC density plot

LOGVAC's skewness is well above 0 (the skewness of a $\mathcal{N}(0,1)$ distribution). The same goes for its kurtosis. This shows that the distribution is asymmetrical and heavy-tailed to the right. Together with the fact that the mean is above the median and the high variance, we can say that LOGVAC is quite spread out, and so that any drawn sample will potentially contain some very influential values. Using an SRSWOR design, whose only required auxiliary information is to know N, we obtained an $n = 80$ units sample s with $\overline{\text{logvac}} = \frac{1}{n}\sum_{i \in s} \text{logvac} \simeq 22.175$ and $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(logvac_i - \overline{logvac})^2 \simeq$ 1 144.121.

It contains observations such as:

- Lege has LOGVAC $= 1 \Rightarrow \hat{B}^{\text{HT}}_{\text{Lege}} = -127.05$
- Rouede has LOGVAC $= 20 \Rightarrow \hat{B}^{\text{HT}}_{\text{Le Faget}} = -13.05$
- Cazeres has LOGVAC $= 214 \Rightarrow \hat{B}^{\text{HT}}_{\text{Cazeres}} = 1\,150.95$

We can also plot the conditional bias of this sample:

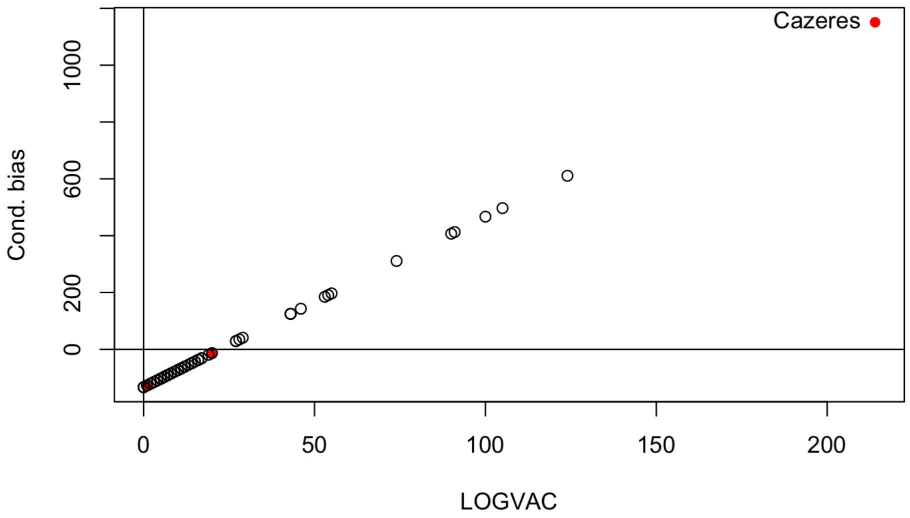


Figure 2: scatterplot of the Conditional Bias against LOGVAC

This small example shows that the conditional bias takes both positive and negative values under an SRSWOR design, and that it linearly increases with the interest variable value. This will help penalizing the influential values such as Cazeres, in the construction of the robust estimator.

2.3 Robust estimator

Now that the conditional bias appears to be a good measure of influence, how to use it in order to obtain robust estimators in a design-based framework? This is actually a question that Beaumont et al. (2013) [BHR13], and Kokic & Bell (1994) [KB94] for the stratified simple random sampling, already asked themselves.

The estimator used in `REVIVALS` and proposed by Beaumont et al. (2013) is the following:

$$\hat{t}_y^{\text{RHT}} = \hat{t}_y^{\text{HT}} - \frac{1}{2}\left(\hat{B}_{\text{min}}^{\text{HT}} + \hat{B}_{\text{max}}^{\text{HT}}\right), \tag{7}$$

where $\hat{B}_{\text{min}}^{\text{HT}} = \hat{B}_{\text{min}}^{\text{HT}}(I_i = 1)$ (resp. $\hat{B}_{\text{max}}^{\text{HT}} = \hat{B}_{\text{max}}^{\text{HT}}(I_i = 1)$) is the minimum (resp. maximum) estimated conditional bias on all the sampled observations.

Nevertheless, this estimator obtained in (7) is in fact obtained from a more general form:

$$\hat{t}_y^{\text{RHT}} = \hat{t}_y^{\text{HT}} - \sum_{i\in S}\hat{B}_i^{\text{HT}}(I_i = 1) + \sum_{i\in S}\psi(\hat{B}_i^{\text{HT}}(I_i = 1)), \tag{8}$$

where ψ is the Huber function, defined as:

$$\psi(x) = \text{sign}(x) \times \min(|x|, c)\ , \tag{9}$$

where $c > 0$, and $\text{sign}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{otherwise} \end{cases}.$

It is interesting to express $\hat{t}_y^{\text{RHT}}$ under the classical weighted sum form $\hat{t}_y^{\text{RHT}} = \sum_{i\in S}\tilde{d}_i y_i$. From (8), if we put everything in a single summation, we have:

$$\hat{t}_y^{\text{RHT}} = \sum_{i\in S} d_i y_i - \hat{B}_i^{\text{HT}} + \psi\left(\hat{B}_i^{\text{HT}}\right) \ \Rightarrow\ \tilde{d}_i = d_i - \frac{\hat{B}_i^{\text{HT}}}{y_i} + \frac{\psi\left(\hat{B}_i^{\text{HT}}\right)}{y_i}$$

This is what is used in practice to get the robust estimator in `REVIVALS`.

Example: Poisson sampling design

Using the fact that $B_{1i} = (d_i - 1)y_i$, we have:

$$\begin{aligned}\tilde{d}_i &= d_i - (d_i - 1) + \frac{\text{sign}\left((d_i - 1)y_i\right)\ \min(|B_{1i}|, c)}{y_i} \\ &= 1 + \frac{\text{sign}\left((d_i - 1)y_i\right)\ \min(|(d_i - 1)y_i|, c)}{y_i}\end{aligned}$$

And therefore,

$$\tilde{d}_i = \begin{cases} 1 + \frac{(d_i - 1)y_i}{y_i} = d_i, & \text{if } |(d_i - 1)y_i| < c \\ 1 + \frac{\text{sign}((d_i - 1)y_i) \times c}{y_i}, & \text{otherwise.} \end{cases}$$

Application

In this application, we estimated the Robust Horvitz-Thompson (RHT) and the HT estimators for each of the three non-stratified designs. For the SRSWOR design, which has equal probabilities, we used the same sample s as in the conditional bias section 2.2. However, the Poisson and the Rejective designs have unequal probabilities. In order to fix their weights, we decided to use an auxiliary variable LOG, which corresponds to the number of dwellings in each commune. It seems logical to believe that the more dwellings in a commune, the more chances for this commune to have a higher number of vacant homes:

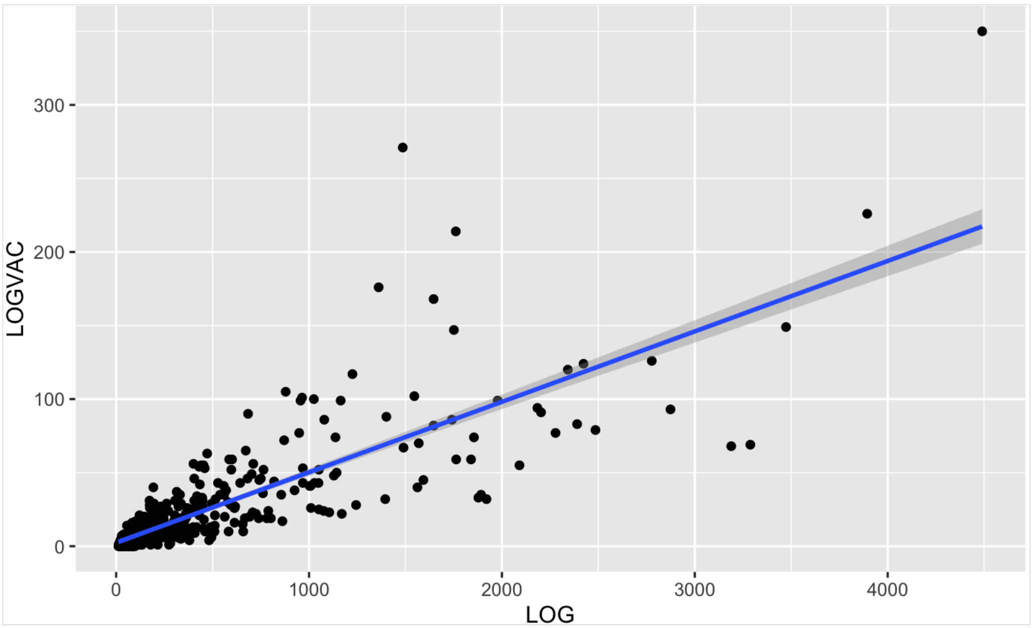


Figure 3: Scatter plot of LOG against LOGVAC

On the scatter plot, we see that there is a linear relationship between LOGVAC and LOG, which confirms the validity of LOG as an auxiliary variable. The correlation coefficient is indeed very close to 1: $\rho(LOG, LOGVAC) \simeq 0.8189$. Therefore, the weights have been fixed to be proportional to LOG:

$$\pi_i = \frac{log_i \times n}{LOG}, \tag{10}$$

where $LOG = \sum_{i \in s} log_i$, and assuming $\forall i, log_i \times n \leq \sum_{i \in s} LOG_i$.

If there exists an i such that $LOG_i \times n > \sum_{i \in s} LOG_i$, then we set $\pi_i = 1$ and recompute all the π_is by subtracting the number of units for which $\pi_i = 1$, to n. In our case, we obtained the following estimates with $n = 78$ for Poisson:

Design	$\hat{t}^{\mathrm{RHT}}_{logvac}$	$\hat{t}^{\mathrm{HT}}_{logvac}$
SRSWOR	11 776.00	12 284.95
Poisson	11 174.90	11 323.90
Rejective	10 875.89	10 958.70

Table 4: Robust HT estimates

The true value (which would be unknown in a real application case) is $t_{\mathrm{LOGVAC}} = 10\,768$. Hence, these are three examples where $\hat{t}^{\mathrm{HT}}_y$ overestimates due to the presence of outliers, whereas $\hat{t}^{\mathrm{RHT}}$ performs better by reducing the effect of influential units. In the case of the SRSWOR design, for instance, Cazères is the only unit whose weight has been modified, going from $d_i = 6.925$ to $\tilde{d}_i = 4.547$. Of course, this is based on a single sample so it serves by no means as general conclusion on the performance of each estimator.

A final remark is that again it wouldn't make sense to compare designs on a single sample. However, by using this method to fix Poisson and Rejective weights, we would, on average, hope to gain some precision with respect to the SRSWOR design.

2.4 Tuning constant

As you could see on (9), there is a constant c that intervenes in the computation of the robust estimator. In order to get $\hat{t}_y^{\mathrm{RHT}}$ itself we don't actually need to compute c. However, it is still interesting to compute it afterwards in some cases. Beaumont et. al [BHR13] proposed a method which consists into choosing c such that it minimizes the largest conditional bias, in absolute value, for the RHT estimator of a unit i. That is solving:

$$\min_c \ \max\{|\hat{B}_i^{\mathrm{RHT}}(c)|; \ i \in s\}, \tag{11}$$

where $\hat{B}_i^{\mathrm{RHT}}(c) = B_i^{\mathrm{HT}} + \Delta(c)$, and $\Delta(c) = \sum_{i \in S}\{\psi(\hat{B}_i^{\mathrm{HT}}; c) - \hat{B}_i^{\mathrm{HT}}\}$

From a more practical point of view, what we did to find c in the R function `tuningconst` (see below) was to minimize[1] the difference between the general equation of the robust estimator (8) and the equation in which the associated tuning constant has been minimized (7):

$$\begin{aligned}(8) - (7) &= \hat{t}_y^{HT} - \sum_{i \in S} B_{1i}^{HT} + \sum_{i \in S} \psi(B_{1i}^{HT}) - \hat{t}_y^{HT} + \frac{1}{2}\left(\hat{B}_{min}^{HT} + \hat{B}_{max}^{HT}\right) \\ &= \sum_{i \in S} \psi(B_{1i}^{HT}) - B_{1i}^{HT} + \frac{1}{2}(\hat{B}_{min}^{HT} + \hat{B}_{max}^{HT})\end{aligned} \tag{12}$$

[1] We used the function `optimize` from the `stats` package

2.5 Robust estimator under the winsorised form

Sometimes it turns out to be easier to write the estimator in (7) under the winsorised form. This is a well-known and widespread common practice used to deal with outliers. INSEE mentions it in their annual publication of the methodology that they follow to treat atypical units of their enterprise surveys [BH17]. In their case, winsorisation is used to deal with companies which have an earnings per employee ratio unusually high compared to the rest of companies from the same sector and with similar size.

The idea behind winsorisation is that any value y_i over a given threshold K is reduced, taking its weight d_i into account. More precisely, if $d_i y_i > K$ then y_i becomes $\tilde{y}_i$. Hence, the winsorised estimator of the total is usually written as: $\hat{t}_s = \sum_{i \in s} d_i \tilde{y}_i$. Though in our case, we will only present the alternative form using modified weights $\hat{t}_s = \sum_{i \in s} \tilde{d}_i y_i$ because that is how it is coded in `REVIVALS`.

Following the literature, we distinguish between two forms of winsorisation. Standard winsorisation on one side, and the winsorisation proposed by Dalén in 1987 [Dal87] and Tambay in 1988 [Tam88] on the other side.

- **Standard**

$$\tilde{d}_i = d_i \frac{min\left(y_i,\, \frac{K}{d_i}\right)}{y_i}. \tag{13}$$

If min $\left(y_i,\, \frac{K}{d_i}\right) = y_i$ (i.e. unit i is not influential), then $\tilde{d}_i = d_i$: the weight of a non-influential unit isn't modified. Nonetheless, the weight of an influential unit is below d_i and can even go below 1.

Remark: If $y_i = 0$ for a given unit i, then this doesn't cause any problem because the contribution of this unit to the total is null, and so in this case we can assign it a random weight $\tilde{d}_i$ such as its initial weight d_i, for instance.

- **Dalén-Tambay**

The idea is the same, but with a slightly different definition:

$$\tilde{d_i} = 1 + (d_i - 1)\frac{min\left(y_i,\, \frac{K}{d_i}\right)}{y_i} \tag{14}$$

As with the standard version, the weight of a non-influential unit is unmodified. The difference between the two forms comes from the fact that Dalén-Tambay ensures that the modified weights cannot be lower than 1.

The choice of the winsorisation K is another key step when applying such methods. The optimal constant K_{opt} which minimises (11) is obtained by solving :

$$\sum_{j \in s} a_j \max\left(0, d_j y_j - K\right) = \frac{\hat{B}^{\text{HT}}_{min} + \hat{B}^{\text{HT}}_{max}}{2}, \tag{15}$$

where $a_j = 1$ in the case of the standard winsorised estimator and $a_j = (d_j - 1)/d_j$ in the case of the DT winsorised estimator. It can be shown (cf. [Fav15]) that a solution to the equation (15) exists under the following conditions:

1. $\pi_{ij} - \pi_i\pi_j \leq 0$;
2. $\frac{1}{2}(\hat{B}^{\text{HT}}_{min} + \hat{B}^{\text{HT}}_{max}) \geq 0$.

Condition 1 is satisfied for most of the frequently used sampling designs such as SRSWOR, STSRSWOR, and Poisson. Condition 2 implies that $\hat{t}^{\text{RHT}}_y$ needs to be lower or equal to $\hat{t}^{\text{HT}}_y$ because a winsorised estimator cannot be grater than a Horvitz-Thompson, by definition.

In the `REVIVALS` package, the `uniroot` function from the package `stats` enables to numerically compute the constant. This function is capable of searching an interval from lower to upper for a root of a given function f. In our case, we wanted to consider all possible thresholds, so we fed it with the interval $K \in [0, \max_{i\in s}(d_i y_i)]$. Then, we defined the functions $g(K) = a_j \max(0, d_j y_j - K),\ \forall j \in s$, and $f(K) = -\sum_{j\in s} g(K) + \frac{1}{2}\left(\hat{B}^{\text{HT}}_{min} + \hat{B}^{\text{HT}}_{max}\right)$. Finally, we control accuracy by setting a convergence tolerance to $tol = 2.913414^{-157}$ (it's 'the smallest positive floating-point number x such that $1 + x \neq 1$' for the machine, to the power 10). The algorithm proceeds with dichotomic search, and convergence is declared either if $\exists\ K^*$such that$f(K^*) = 0$, or if change in K for one step of the algorithm is less than tol. If there's no convergence after 10 000 iterations, the function returns an approximation.

K_{opt} is the threshold above which large values are reduced, and so in a way it controls the average error of our estimator. This is why a good choice is important.

3 Revivals

3.1 Initial structure

The very first thing that I had to do when I was introduced to REVIVALS was to get familiar with the package, and make the association between every function and the theory. The way it had been structured so far by C. Favre-Martinoz et al. (cf. [Fav+16]) was to have 6 separate functions:

- HTcondbiasest: a function giving the estimation of the conditional bias of the Horvitz-Thompson estimator ;
- robustest: a function calculating the robust Horvitz-Thompson estimator proposed in (7);
- tuningconst: a function which calculates the tuning constant associated to the obtained robust estimator ;
- robustweights: a function which calculates the weights associated to the two winsorized estimators detailed in section 2.5, as well as those associated to the BHR estimator (7) ;
- determinconstws and determinconstwDT: two functions computing the winsorisation constants associated to the standard winsorized estimator and the Dalen-Tambay winsorized estimator respectively.

For each of these functions, apart from those calculating a constant, had also been coded its stratified counterpart (the strata_ prefix was added in front of the name), which summed up to a total of 9 functions.

The arguments could vary depending on the function, but the idea was to give the user the choice of the sampling design. Three possibilities were offered:

```
method = c('si','poisson','rejective').
```

Will now be presented the enhancements that have been made.

3.2 Robustness

In order to improve the package robustness, we started by carrying out some tests to find out cases for which the functions couldn't handle the data.

Here is an exhaustive list of all the warnings that have been added:

- If both d_i and π_i are specified, a warning notifying that d_i is redundant and only π_i is being used appears ;
- During an estimation with rejective sampling design, we don't want to constrain the user so we simply decided to add a warning telling him to make sure that D is large enough and N/D is bounded ;
- A message warns the user when the estimation method is wrongly specified or multiple methods are specified. In both cases the method is set to 'si' by default.
- There is a warning when one of the interest variables contains negative values. The reason why we didn't make this a stop is because we saw that in some cases, it could be that the $\hat{B}^{\text{HT}}_{\text{min}} + \hat{B}^{\text{HT}}_{\text{max}}$ from the robust estimator equation $\hat{t}^{\text{RHT}}_y = \hat{t}^{\text{HT}}_y - \frac{1}{2}\left(\hat{B}^{\text{HT}}_{\text{min}} + \hat{B}^{\text{HT}}_{\text{max}}\right)$ is negative, in which case we would be adding something to the HT estimator instead of subtracting from it. Hence, we decided not to put any constraint on the y_i, and to only check the condition: $\frac{1}{2}(\hat{B}_{min} + \hat{B}_{max}) \geq 0 \Leftrightarrow \hat{B}_{max} \geq -\hat{B}_{min}$, giving the existence of the tuning constant. Note that this is only true for the winsorised estimators.

Here is an exhaustive list of all the stops that have been added:

- We make sure that none of the interest variables contains any missing value ;
- We stop the code when π_i (or d_i) has a different number of rows than the data file ;
- As explained before, we check that $\frac{1}{2}(\hat{B}_{min} + \hat{B}_{max}) \geq 0 \Leftrightarrow \hat{B}_{max} \geq -\hat{B}_{min}$. Functions `determinconsts` and `determinconstwDT` giving the winsorisation constants do not work otherwise.

3.3 New features

We implemented two major new features:

- **Addition of an identifier**

  One of the first things we noticed during the testing phase was that we were lacking an identifier. When looking at the table of conditional biases, for instance, it seemed obvious wanting to make the association between individuals and the displayed values. Hence we decided to add an argument 'id' to the `condbias` function, through which the user could specify the column that would be used as a unique key. This keeps a link between the input data set (which in practice only contains some surveyed observations) and the data frame containing the computed conditional biases. If the user chooses to remerge the result with the original data set, this is not necessary so no identifier is added. But if 'remerge=FALSE' and no column is specified, an id numbering every row is automatically added at the beggining of the result in order to keep track and remerge the output later on if wished.

  Also, if the argument 'id' is set to 'none', then nothing is checked or added. This was initially conceived to be a tool to omit all the warning messages when the function `condbias` was used internally in another function (in `strata_condbias`, for instance), but turned out to be a useful option in some cases.

- **New function: wrapper**

  Executing each of the package functions individually seemed like a bit too heavy for a standard user which would only want to quickly obtain the value of the robust estimator for a given interest variable, for example. Therefore, we thought about wrapping everything up in a single function (named `wrapper`), which would output a list of two tables:

  - ◇ the first table summarizes the main results: for every type of estimation, and every interest variable, it gives the robust HT estimation, the classical HT estimation, the relative difference between the two (with no absolute value), and the number of modified weights.

- ◇ the second table is more detailed. For every individual in the sample, we have: its initial weight, its interest variable value, its conditional bias, its new weight for the robust estimation, as well as an indicator allowing to quickly see for which observations has the weight been modified.

The wrapper function integrates all the sampling designs, including the stratified ones, as well as the 3 types of estimation: BHR, standard winsorisation and Dalén-Tambay winsorisation. Even with multiple interest variables, sampling designs, and all the types of estimation selected, the execution part doesn't take long (just a few seconds at most), so there were no worries on this side.

For more details, everything is illustrated in part 4.

3.4 Other modifications

Here is a list of all the other minor updates:

- Deleted argument tailleseq from `tuningconst` because unused ;
- Added the option to either specify the first order inclusion probabilities π_i, or its inverse d_i.
- Minor syntax tweeks to better understand some outputs and improve code readability. Here are some of them:
  - ◇ Was added a 'stratum' column in the output of `strata_robustest` to correctly associate the estimations with the strata
  - ◇ Were added some prints during the execution of `strata_robustest` to be more precise about to which stratum did the warning / stop messages belong ;
  - ◇ Were added relevant prints when the sum of the inclusion probabilities is not equal to N in `condbiasest` and `strata_condbiasest`, as well as in `robustest` and `strata_robustest` to show the values of D and N/D to the user.
- All the documentation has also been accordingly modified to account for all of these changes.

3.5 Article

The R Journal is an online journal which has been publishing research articles in statistical computing since 2009. With the publication of our package, we decided to write an article which would be kind of a user manual, in order to submit it to the R Journal. Inside is explained the motivation, along with some illustrations behind every function of the package.

Writing the article, while making sure we complied with their format conditions, was also one of my main tasks during this Master thesis.

The article will most likely be submitted by the end of September 2020, at the end of my working period.

4 Illustration with the wrapper function

4.1 Non-stratified designs

For the fixed-size designs (SRSWOR, and Rejective sampling), we start by declaring $n = 80$. We also remind you that $N = 554$.

We then fix the respective first order probabilities: $\pi_i = \frac{80}{554} \simeq 0.144$ for SRSWOR, and $\pi_i = \frac{log_i \times 80}{197\,314}$ for Poisson and Rejective designs, in order to draw some samples. To do so, we used some functions from the **`sampling`** package, from Tillé and Matei [TM16]:

- **`inclusionprobabilities`**: 'computes the first-order inclusion probabilities from a vector of positive numbers (in our case the *LOG* variable) for a proportional-to-size sampling design'
- `srswor`, `UPpoisson` and `UPmaxentropy`: draws each of the three samples (note that with `UPpoisson`, the size is random).

Here is how the syntax of the wrapper function looks like (see the full R code in the Appendix):

```
wrapper(data = ech, varname = c("LOGVAC"),
        gn = N,
        est_type = c("BHR", "standard", "DT"),
        method = "si",
        pii = ech$piks, id = "CODE_N")
```

The *method* argument can be set to 'poisson' or 'rejective' to change the design. Also note that we could add some other interest variables in the vector *varname*, although for the purpose of this example we will keep it simple and stick with LOGVAC solely. *est_ type* stands for estimator type, and here we indicate that we want the output to display results for all the three possible types: Beaumont Haziza Ruiz-Gazen, standard winsorisation, and Dalén-Tambay winsorisation. Moreover, CODE_N is a unique identifier of every observation, so we use it as an *id*. Finally, *ech* (which comes from the French 'échantillon'), is our sample data and contains a column named *piks* with the first-order probabilities.

4.1.1 Summary tables

Here are the summary tables for each of the three sampling designs:

	est_type	var	RHT	tuning_const	HT	rel_diff	nb_modif_weights
1	BHR	LOGVAC	11 776.00	642.00	12 284.95	-4.14	1
2	standard	LOGVAC	11 776.00	973.00	12 284.95	-4.14	1
3	DT	LOGVAC	11 776.00	887.10	12 284.95	-4.14	1

Table 5: Summary table for SRSWOR

	est_type	var	RHT	tuning_const	HT	rel_diff	nb_modif_weights
1	BHR	LOGVAC	11 174.90	239.78	11 323.94	-1.32	5
2	standard	LOGVAC	11 174.90	292.60	11 323.94	-1.32	5
3	DT	LOGVAC	11 174.90	271.66	11 323.94	-1.32	6

Table 6: Summary table for Poisson sampling

	est_type	var	RHT	tuning_const	HT	rel_diff	nb_modif_weights
1	BHR	LOGVAC	10 875.89	238.57	10 958.70	-0.76	2
2	standard	LOGVAC	10 875.89	402.24	10 958.70	-0.76	3
3	DT	LOGVAC	10 875.89	393.87	10 958.70	-0.76	3

Table 7: Summary table for Rejective sampling

The estimated values for the RHT and HT estimators are the same than in part 2.3. One thing we notice at first glance, though, is that in every table, what varies is the associated tuning constant and the number of modified weights, while the value of the RHT estimator stays the same. The types of estimation are indeed just three different ways of re-parametrization. This underlines the fact that there is no unicity of the weighting system.

Also, the relative difference column indicates the percentage change between the HT estimator, taken as a reference measure, and the RHT estimator:

$$\text{rel\_diff} = \frac{t_y^{\text{RHT}} - t_y^{\text{HT}}}{t_y^{\text{HT}}} \times 100$$

We did not use absolute values in order to indicate the change direction.

4.1.2 Detailed tables

In the Appendix you will find the header of the detailed tables for each of the three sampling designs, which is the second output of the wrapper function (see Tables 13, 14 and 15). On each of them, we can see that the new weights for Dalén-Tambay are always equal or greater to 1. In fact, the only ones that are exactly equal to 1 are those which were set to 1 during the sampling process explained in the Application of 2.3. This verifies the theoretical property of the DT weights.

Still, what is also interesting to look at is the list of units that we obtain by filtering all the rows for which modified_LOGVAC_BHR, modified_LOGVAC_standard, and modified_LOGVAC_DT are 'TRUE' in any of these three tables. This corresponds to the units whose weight has been modified, and we present them in the tables down below:

- For SRSWOR, the observation which is modified is always the same for the three weight types:

est_type	CODE_N	Commune	LOG	LOGVAC	init_weight	modif_weight
BHR / std / DT	31135	Cazeres	1 761	214	6.925	4.547

Table 8: Modified observation(s) for SRSWOR

- For Poisson sampling we have:

est_type	CODE_N	Commune	LOG	LOGVAC	init_weight	modif_weight
BHR / std / DT	31083	Boussan	121	16	19.602	15.987 / 18.287 / 17.112
BHR / DT	31286	Lavelanet-de-C.	231	27	10.268	9.881 / 10.082
BHR	31235	Guran	65	7	36.490	35.255
BHR / std / DT	31521	Saleich	197	27	12.040	9.881 / 10.837 / 10.226
BHR / std / DT	31396	Nailloux	472	63	5.025	4.806 / 4.644 / 4.454
std	31042	Bagneres-de-B.	4 490	350	1.000	0.836
std / DT	31375	Montesquieu-V.	1 361	176	1.743	1.662 / 1.658
DT	31135	Cazeres	1 761	214	1.347	1.327

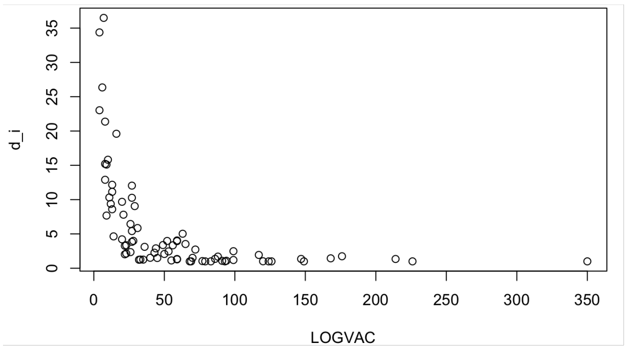
Table 9: Modified observations for Poisson sampling

- For Rejective sampling we have:

est_type	CODE_N	Commune	LOG	LOGVAC	init_weight	modif_weight
BHR / std / DT	31146	Cires	39	7	60.817	55.438 / 57.463 / 56.342
BHR / std / DT	31590	Binos	22	4	107.812	96.526 / 100.560 / 98.555
std / DT	31390	Montrejeau	1 486	271	1.596	1.484 / 1.543

Table 10: Modified observations for Rejective sampling

For simple random sampling without replacement, LOG isn't used to set the weights, so the only factor that affects the decision to modify them or not is the value of LOGVAC: the higher it is, the higher the conditional bias, and so the more chances for a weight to be modified. However, the auxiliary variable LOG comes into play for Poisson and Rejective designs. For any i, the lower the log_i, the lower the π_i and so the higher the d_i. Let us plot them in the example of our Poisson sample:



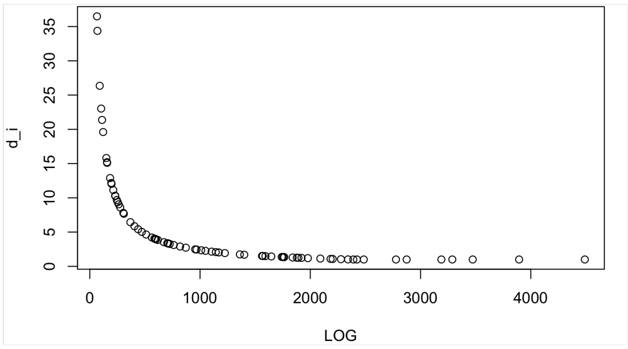


Figure 4: Scatterplot of d_i versus LOGVAC

Figure 5: Scatterplot of d_i versus LOG

It is not surprising to have a $y = \frac{1}{x}$ shaped curve as $d_i = \frac{1}{\pi_i} = \frac{LOG}{n} \times \frac{1}{log_i}$. This explains why for Poisson and Rejective, we see more modified weights than for an equal probabilities design like SRSWOR. More precisely, there are two effects in the conditional bias for Poisson and Rejective, we have either a week chance of being selected (low value of LOG), or a high LOGVAC value. Globally, all units in Table 9 fall under one of these two categories.

It leads us to believe that a design with unequal probabilities, meaning whose weights depend on an auxiliary variable, will potentially often have a greater number of modified weights than a design with equal probabilities like SRSWOR.

Another aspect worth noticing is the differences between the new weights for every type of estimation. We plotted the new weights used for the winsorised standard estimator against those used for the BHR estimator, as well as those from Dalén-Tambay against BHR.

In red are highlighted all the units whose weight has been modified. We can see that despite being different units, the total number of those which are modified does not vary that much in our example.

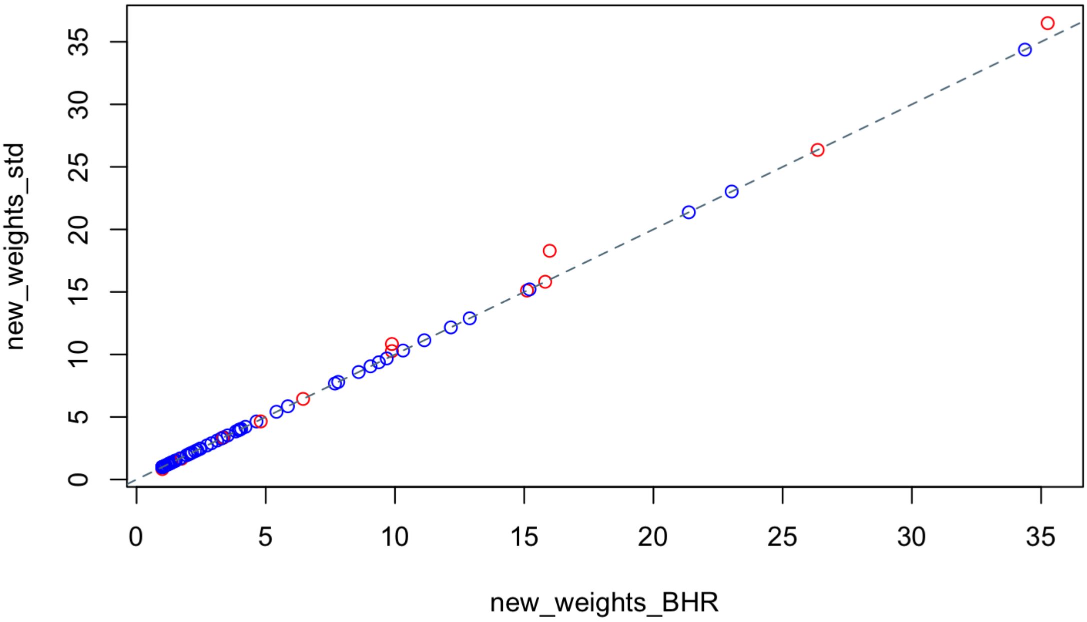


Figure 6: Scatterplot of $\tilde{d}_i^{std}$ against $\tilde{d}_i^{BHR}$

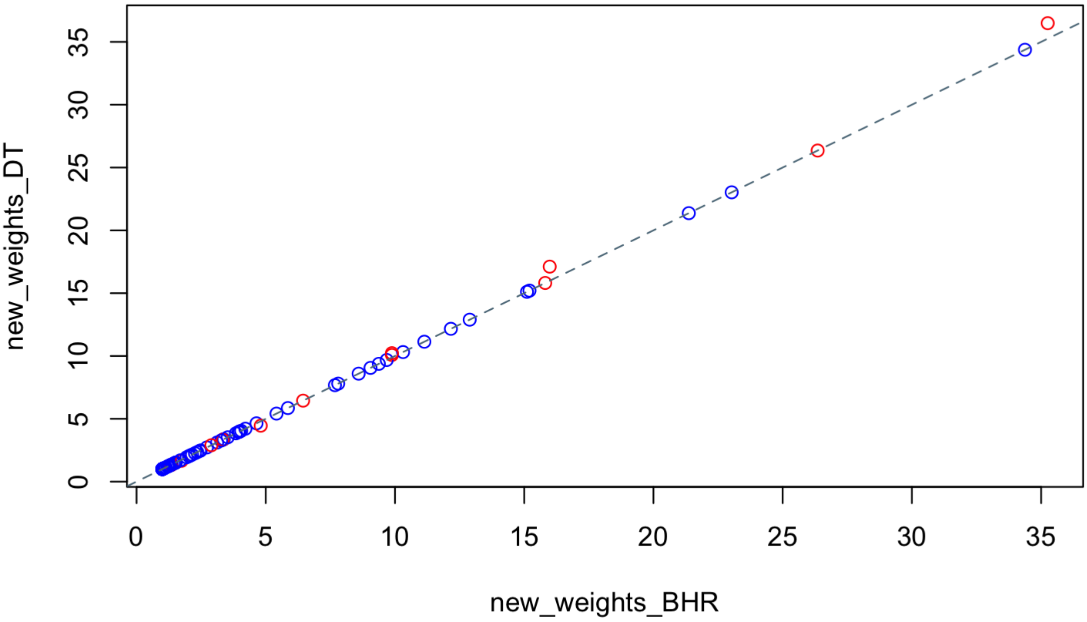


Figure 7: Scatterplot of $\tilde{d}_i^{DT}$ against $\tilde{d}_i^{BHR}$

4.2 Stratified designs

The stratified design samples are drawn similarly to the non-stratified ones. Here the population is partitioned into $H = 4$ strata: $U = \cup_{h=1}^{H=4} U_h$ and $N = \sum_{h=1}^{H=4} N_h$ with $N_1 = 221$, $N_2 = 169$, $N_3 = 110$ and $N_4 = 54$.

We also start by defining the first-order inclusion probabilities: $\pi_i = \frac{n_h}{N_h}$ for stratified simple random sampling, and $\pi_i = \frac{log_i \times n_h}{LOG_h}$, where $LOG_h = \sum_{i \in U_h} log_i$ for Poisson and Rejective designs. This enables us to draw a sample $s = \cup_{h=1}^{H=4} s_h$ of size n_h in each case. We remind that the draw in one stratum is independent from the draw in any other stratum.

In this case, we also used the function `strata` from the `sampling` package. The argument 'method' also allows to choose the design under which the sample units are selected.

Lastly, for SRSWOR we decided to use proportional allocation: $n_h = \frac{n \times N_h}{N}$. This yields: $n_1 = 32$, $n_2 = 24$, $n_3 = 16$ and $n_4 = 8$.

Again, the syntax for the wrapper function is similar to what we had in the non-stratified section:

```
wrapper(data = ech,
        varname = c("LOGVAC"),
        strataname = "Stratum",
        gnh = N_h,
        est_type = c("BHR", "standard", "DT"),
        method = "si", # could aslo be "poisson" or "rejective"
        pii = ech$piks,
        id = "CODE_N")
```

The main difference is that we replaced N by the vector of sizes N_h, and that we added the argument *strataname = 'Stratum'* to indicate the name of the column containing the information about the stratum of every observation. Again, we only focus on LOGVAC as the only interest variable, *ech* is our data frame containing all the sample data, we still want the 3 types of estimations in est_type, and id is our unique identifier. Please see the full R code in the Appendix for further information.

4.2.1 Summary tables

Here are the summary tables for two of the three stratified sampling designs:

stratum	est_type	var	RHT	tuning_const	HT	rel_diff	nb_modif_weights
1	BHR	LOGVAC	926.53	326.01	946.16	-2.07	2
2	BHR	LOGVAC	1 615.65	326.01	1654.79	-2.37	2
3	BHR	LOGVAC	3 513.20	326.01	3602.50	-2.48	1
4	BHR	LOGVAC	4 186.00	326.01	4 191.75	-0.14	2
TOTAL	BHR	LOGVAC	10 241.38		10 395.20	-1.48	7
1	standard	LOGVAC	926.53	804.24	946.16	-2.07	1
2	standard	LOGVAC	1 615.65	804.24	1 654.79	-2.37	2
3	standard	LOGVAC	3 513.20	804.24	3 602.50	-2.48	1
4	standard	LOGVAC	4 186.00	804.24	4 191.75	-0.14	1
TOTAL	standard	LOGVAC	10 241.38		10 395.20	-1.48	5
1	DT	LOGVAC	926.53	798.54	946.16	-2.07	2
2	DT	LOGVAC	1 615.65	798.54	1 654.79	-2.37	2
3	DT	LOGVAC	3 513.20	798.54	3602.50	-2.48	1
4	DT	LOGVAC	4 186.00	798.54	4 191.75	-0.14	1
TOTAL	DT	LOGVAC	10 241.38		10 395.20	-1.48	6

Table 11: Summary table for STSRSWOR

stratum	est_type	var	RHT	tuning_const	HT	rel_diff	nb_modif_weights
1	BHR	LOGVAC	901.86	479.50	946.16	-4.68	2
2	BHR	LOGVAC	1 558.12	479.50	1654.79	-5.84	2
3	BHR	LOGVAC	3 326.38	479.50	3602.50	-7.66	2
4	BHR	LOGVAC	3 740.38	479.50	4191.75	-10.77	5
TOTAL	BHR	LOGVAC	9 526.73		10 395.20	-8.35	11
1	standard	LOGVAC	901.86	574.90	946.16	-4.68	2
2	standard	LOGVAC	1 558.12	574.90	1 654.79	-5.84	2
3	standard	LOGVAC	3 326.38	574.90	3 602.50	-7.66	2
4	standard	LOGVAC	3 740.38	574.90	4 191.75	-10.77	5
TOTAL	standard	LOGVAC	9 526.73		10 395.20	-8.35	11
1	DT	LOGVAC	901.86	562.54	946.16	-4.68	2
2	DT	LOGVAC	1 558.12	562.54	1 654.79	-5.84	2
3	DT	LOGVAC	3 326.38	562.54	3 602.50	-7.66	2
4	DT	LOGVAC	3 740.38	562.54	4 191.75	-10.77	5
TOTAL	DT	LOGVAC	9 526.73		10 395.20	-8.35	11

Table 12: Summary table for Stratified Poisson sampling

We see that the relative difference is negative everywhere: the presence of at least one influential value in every stratum pulls the RHT estimator down compared to the HT estimator.

Also, the rows in blue indicating the totals are purely informative. If one wishes to have an estimation of the total on the whole population, he should instead use the functions presented for the non-stratified case. Summing the total in every stratum produces an estimator which is potentially biased, so one should proceed with caution.

Another remark is that on the second table (Stratified Poisson sampling), the number of modified weights in each stratum is always the same when we compare the three types of estimation. Still, this shouldn't lead us to think that it is the same eleven observations that are modified every time. As we saw in the non-stratified situations, sometimes using a BHR estimation will consider a unit influential that any of the two winsorisation method won't. This is the *raison d'être* of the second output of the wrapper function, containing the detailed table (their headers are presented in the Appendix).

5 Conclusion

The work done during this master thesis has contributed to improve the package `REVIVALS`. We progressively refined it by implementing new stops and warning conditions that make it more robust. In addition, we added some useful features that turn it into a more user-friendly package, specially with the ease of use brought by the wrapper function. By executing a single function, one can not only see the robust HT estimator of the total, but also compare it to the classical HT estimator, and even look for the observations whose weight has been modified if he wants some more detail.

The numerical example with the *rec99htegne* data set has enabled us to see a real application case where the robust Horvitz-Thompson estimator turns out to be advantageous, by detecting outliers and reducing their weights. We remind, however, that this does not happen all the time: we didn't show, for instance, what would have happened if we'd had a left-hand side tail on the distribution (or tails on both sides). In any case, the general goal is to try being efficient on all samples, on average.

Finally, extensions of this work could consist into trying to implement Kokic and Bell methods to our R package. It provides a way to determine the threshold of the winsorised estimator for a stratified simple random sampling without replacement. This method is still extensively used nowadays when we have some historical auxiliary information at our disposal (data from a previous survey for example), and it has recently been extended to the Poisson framework (cf. [DF18]) by Deroyon and Favre-Martinoz (2018). Its scope is thus close to the one implemented in the package and so its addition would both be a nice complement and a good benchmark.

6 Appendix

6.1 R code

```
rec99htegne <- read.csv(rec99htegne.csv)
# Define the poulation size and the sample size
N = nrow(rec99htegne)
n = 80


# Descriptive stats LOGVAC
summary(rec99htegne$LOGVAC)
skewness(rec99htegne$LOGVAC)
kurtosis(rec99htegne$LOGVAC)
var(rec99htegne$LOGVAC)
plot(density(rec99htegne$LOGVAC), xlab="LOGVAC", main="")


# LOG / LOGVAC
ggplot(rec99htegne, aes(x=LOGVAC, y=LOG)) + geom_point()
    + geom_smooth(method="lm")
cor(rec99htegne$LOGVAC, rec99htegne$LOG)
```

6.1.1 Code for non-stratified designs

```
set.seed(1906)
# Drawing for every sampling design: non-stratified
  # SRSWOR
pii <- rep(n/N, n)
s <- srswor(n, N)
  # Poisson
pii <- inclusionprobabilities(rec99htegne$LOG, n)
s <- UPpoisson(pii)
pii <- pii[s==1]
```

```
  # Rejective
pii <- inclusionprobabilities(rec99htegne$LOG, n)
s <- UPmaxentropy(pii)
pii <- pii[s==1]

# Get the sample data
ech <- rec99htegne[s==1,]
ech$piks <- pii

# wrapper
tb11 <- wrapper(data = ech,
                varname = c("LOGVAC"),
                gn = N,
                est_type = c("BHR", "standard", "DT"),
                method = "si",
                pii = ech$piks,
                id = "CODE_N")
tb12 <- wrapper(data = ech,
                varname = c("LOGVAC"),
                gn = N,
                est_type = c("BHR", "standard", "DT"),
                method = "poisson",
                pii = ech$piks,
                id = "CODE_N")
tb13 <- wrapper(data = ech,
                varname = c("LOGVAC"),
                gn = N,
                est_type = c("BHR", "standard", "DT"),
                method = "rejective",
                pii = ech$piks, id = "CODE_N")
```

```
# Plots

    # LOGVAC / condbias

z <- tb11[[2]]$CODE_N %in% c(31290,31179,31135)

plot(tb11[[2]]$LOGVAC, tb11[[2]]$condbias_LOGVAC,

     xlab="LOGVAC", ylab="Cond.␣bias",

     pch=ifelse(z,16,1),

     col=ifelse(z,'red','black')); abline(

       h=0, v=0); text(

       x=tb11[[2]][tb11[[2]]$CODE_N=="31135","LOGVAC"],

       y=tb11[[2]][tb11[[2]]$CODE_N=="31135","condbias_LOGVAC"],

       labels="Cazeres",

       pos="2")


    # LOG vs di / LOGVAC vs di

tb12LOG <- merge(tb12[[2]], rec99htegne, by="CODE_N")

plot(tb12LOG$LOG, tb12LOG$init_weight, xlab="LOG", ylab="$d_i$")

plot(tb12[[2]]$LOGVAC, tb12[[2]]$init_weight, xlab="LOGVAC", ylab="d_i")


    # BHR vs std / BHR vs DT

z <- tb12[[2]]$new_weights_LOGVAC_BHR!=tb12[[2]]$new_weights_LOGVAC_standard

plot(tb12[[2]]$new_weights_LOGVAC_BHR, tb12[[2]]$new_weights_LOGVAC_standard,

     xlab="new_weights_BHR",

     ylab="new_weights_std",

     col=ifelse(z,'red','blue')); abline(0, 1, col="lightskyblue4", lty=2)


z <- tb12[[2]]$new_weights_LOGVAC_BHR != tb12[[2]]$new_weights_LOGVAC_DT

plot(tb12[[2]]$new_weights_LOGVAC_BHR, tb12[[2]]$new_weights_LOGVAC_DT,

     xlab="new_weights_BHR",

     ylab="new_weights_DT",

     col=ifelse(z,'red','blue')); abline(0, 1, col="lightskyblue4", lty=2)
```

6.1.2 Code for stratified designs

```
# Population size and sample size in each stratum (prop. allocation)

N_h = as.vector(table(rec99htegne$stratlog))

nh_prop = vector()

for (i in 1:length(N_h)){ nh_prop[i] = round(n * N_h[i] / N) }


set.seed(1906)

# Drawing for every sampling design: stratified

  # STSRSWOR

pii_strata <- as.data.frame(cbind(c(1:length(nh_prop)), nh_prop / N_h))

names(pii_strata) = c("strata", "piks")

datas <- merge(rec99htegne, pii_strata, by.x="stratlog", by.y="strata")

st = strata(datas, stratanames=c("stratlog"), size=nh_prop, method="srswor")


  # STPoisson

pii_strata <- c()

for (i in 1:nrow(rec99htegne)) {

  pii_strata[i] <- rec99htegne[i, "LOG"] * nh_prop[rec99htegne[i, "stratlog"]]

  / sum(rec99htegne[rec99htegne[,"stratlog"]==rec99htegne[i,"stratlog"],"LOG"])

}

datas <- rec99htegne

datas$piks <- pii_strata

st <- strata(datas,

             stratanames=c("stratlog"), size=nh_prop,

             method="poisson",

             pik=datas$piks,

             description=TRUE)


# Get the sample data

ech <- getdata(datas, st)
```

```
# wrapper
tb21 <- wrapper(data = ech,
        varname = c("LOGVAC"),
        strataname = "Stratum",
        gnh = N_h,
        est_type = c("BHR", "standard", "DT"),
        method = "si",
        pii = ech$piks,
        id = "CODE_N")

tb22 <- wrapper(data = ech,
        varname = c("LOGVAC"),
        strataname = "Stratum",
        gnh = N_h,
        est_type = c("BHR", "standard", "DT"),
        method = "poisson",
        pii = ech$piks,
        id = "CODE_N")
```

6.2 Detailed tables from wrapper

(see next two pages)

CODE_N	init_weight	LOGVAC	condbias	new_weights	modifed	new_weights	modifed	new_weights	modifed
			LOGVAC	LOGVAC_BHR	LOGVAC_BHR	LOGVAC_standard	LOGVAC_standard	LOGVAC_DT	LOGVAC_DT
31342	6.92	10.00	-73.05	6.92	FALSE	6.92	FALSE	6.92	FALSE
31020	6.92	90.00	406.95	6.92	FALSE	6.92	FALSE	6.92	FALSE
31086	6.92	5.00	-103.05	6.92	FALSE	6.92	FALSE	6.92	FALSE
31002	6.92	7.00	-91.05	6.92	FALSE	6.92	FALSE	6.92	FALSE
31264	6.92	11.00	-67.05	6.92	FALSE	6.92	FALSE	6.92	FALSE

Table 13: Detailed table for SRSWOR

CODE_N	init_weight	LOGVAC	condbias	new_weights	modifed	new_weights	modifed	new_weights	modifed
			LOGVAC	LOGVAC_BHR	LOGVAC_BHR	LOGVAC_standard	LOGVAC_standard	LOGVAC_DT	LOGVAC_DT
31342	15.81	10.00	148.12	15.81	FALSE	15.81	FALSE	15.81	FALSE
31083	19.60	16.00	297.64	15.99	TRUE	18.29	TRUE	17.11	TRUE
31027	34.37	4.00	133.50	34.37	FALSE	34.37	FALSE	34.37	FALSE
31052	4.21	20.00	64.26	4.21	FALSE	4.21	FALSE	4.21	FALSE
31206	6.45	26.00	141.58	6.45	FALSE	6.45	FALSE	6.45	FALSE

Table 14: Detailed table for Poisson sampling

CODE_N	init_weight	LOGVAC	condbias	new_weights	modifed	new_weights	modifed	new_weights	modifed
			LOGVAC	LOGVAC_BHR	LOGVAC_BHR	LOGVAC_standard	LOGVAC_standard	LOGVAC_DT	LOGVAC_DT
31342	15.81	10.00	12.41	15.81	FALSE	15.81	FALSE	15.81	FALSE
31083	19.60	16.00	160.15	19.60	FALSE	19.60	FALSE	19.60	FALSE
31027	34.37	4.00	-7.17	34.37	FALSE	34.37	FALSE	34.37	FALSE
31052	4.21	20.00	-46.23	4.21	FALSE	4.21	FALSE	4.21	FALSE
31206	6.45	26.00	19.18	6.45	FALSE	6.45	FALSE	6.45	FALSE

Table 15: Detailed table for Rejective sampling

CODE_N	init_weight	stratum	LOGVAC	condbias LOGVAC	new_weights LOGVAC_BHR	modifed LOGVAC_BHR	new_weights LOGVAC_standard	modifed LOGVAC_standard	new_weights LOGVAC_DT	modifed LOGVAC_DT
31039	6.91	1	2.00	-13.91	6.91	FALSE	6.91	FALSE	6.91	FALSE
31415	6.91	1	1.00	-20.01	6.91	FALSE	6.91	FALSE	6.91	FALSE
31002	6.91	1	7.00	16.58	6.91	FALSE	6.91	FALSE	6.91	FALSE
31171	6.91	1	2.00	-13.91	6.91	FALSE	6.91	FALSE	6.91	FALSE
31536	6.91	1	1.00	-20.01	6.91	FALSE	6.91	FALSE	6.91	FALSE
31216	6.91	1	9.00	28.77	6.91	FALSE	6.91	FALSE	6.91	FALSE

Table 16: Detailed table (STSRSWOR)

CODE_N	init_weight	stratum	LOGVAC	condbias LOGVAC	new_weights LOGVAC_BHR	modifed LOGVAC_BHR	new_weights LOGVAC_standard	modifed LOGVAC_standard	new_weights LOGVAC_DT	modifed LOGVAC_DT
31134	11.95	1	2.00	21.91	11.95	FALSE	11.95	FALSE	11.95	FALSE
31415	11.27	1	1.00	10.27	11.27	FALSE	11.27	FALSE	11.27	FALSE
31027	5.72	1	4.00	18.87	5.72	FALSE	5.72	FALSE	5.72	FALSE
31256	6.92	1	4.00	23.68	6.92	FALSE	6.92	FALSE	6.92	FALSE
31019	11.60	1	0.00	0.00	11.60	FALSE	11.60	FALSE	11.60	FALSE
31123	4.70	1	7.00	25.87	4.70	FALSE	4.70	FALSE	4.70	FALSE

Table 17: Detailed table (STPoisson)

Bibliography

[Dal87] J. Dalén. *Practical estimators of a population total which reduce the impact of large observations.* Statistiska centralbyrån, 1987.

[Tam88] J.-L. Tambay. “An Integrated Approach for the Treatment of Outliers in Sub-Annual Economic Surveys”. In: *Proceedings of the Section on Survey Research Methods: American Statistical Association.* 1988, pp. 229–234.

[KB94] P.N. Kokic and P.A. Bell. “Optimal winsorizing cutoffs for a stratified finite population estimator”. In: *Journal of Official Statistics* 10.4 (1994), p. 419.

[MMM99] J.-L. Moreno-Rebollo, A. Muñoz-Reyes, and J. Muñoz-Pichardo. “Miscellanea. influence diagnostic in survey sampling: conditional bias”. In: *Biometrika* 86.4 (1999), pp. 923–928.

[BHR13] J.-F. Beaumont, D. Haziza, and A. Ruiz-Gazen. “A unified approach to robust estimation in finite population sampling”. In: *Biometrika* 100.3 (2013), pp. 555–569.

[Fav15] C. Favre-Martinoz. “Estimation robuste en population finie et infinie”. PhD thesis. 2015.

[Fav+16] C. Favre-Martinoz, A. Ruiz-Gazen, J.-F. Beaumont, and D. Haziza. “Robustness in survey sampling using the conditional bias approach with R implementation”. In: *Convegno della Società Italiana di Statistica.* Springer. 2016, pp. 3–13.

[TM16] Y. Tillé and A. Matei. “Package ‘sampling’”. In: *Survey Sampling. Kasutatud* 23 (2016), p. 2017. URL: `http://cran.r-%20project.org/src/contrib/Descriptions/sampling.html`.

[BH17] J.-M. Béguin and O. Haag. “Méthodologie de la statistique annuelle d’entreprises. description du système «ésane»”. In: (2017).

[DF18] T. Deroyon and C. Favre-Martinoz. “Comparison of the conditional bias and Kokic and Bell methods for Poisson and stratified sampling”. In: *Survey Methodology* 44.2 (2018), pp. 309–338.