United States
Department of
Agriculture

Introducing a New Integer Calibration Procedure

Luca Sartore
Kelly Toppin
Clifford Spiegelman

National
Agricultural
Statistics
Service

Research and
Development Division
Washington DC 20250

RDD Research Report
Number RDD-16-STS

July 2016

TABLE OF CONTENTS

Introducing a New Integer Calibration Procedure

Luca Sartore[*], Kelly Toppin[†] and Clifford Spiegelman[‡]

Abstract

The USDA's National Agricultural Statistics Service (NASS) conducts a census of agriculture every 5 years, in years ending in 2 and 7. The census describes the characteristics of U.S. farms and the people who operate them. It is the only source of uniform, comprehensive data for every state and county. To adjust for undercoverage, nonresponse and misclassification, NASS adjusts the weights on the responding records using a capture-recapture methodology. However, the weights need to be further refined through a calibration process so that the census estimates agree with known population values. Some of the census estimates, such as demographic estimates, are included among the targets so that these estimates are not distorted during calibration. Current NASS calibration methodology is robust but often fails to match all target simultaneously. In this article, we describe a new calibration procedure based on L_1-norm relative error. We present the results of a simulation study designed to investigate the consistency and efficiency of the estimators. The calibration estimator can match more targets simultaneously thus providing a foundation for a better methodology.

KEY WORDS: Calibration; Weights; Rounding; Integer optimization.

[*]National Institute of Statistical Sciences.
[†]National Agricultural Statistics Service, United States Department of Agriculture.
[‡]Texas A&M University.

1 INTRODUCTION

The idea of calibration was first introduced by Lemel (1976). Lemel's idea gained prominence in the statistical world after Deville (1988) and Deville and Särndal (1992) generalized it. Calibration ensures that the "sample" totals match some previously known totals of the population.

Consider a finite population U with N units numbered by index, $i \in \{1, \ldots, N\}$ and with p number of variables. Let x_{ji} be a value of the variable of interest, x_j, with $j \in \{1, \ldots, p\}$, for the i-th population member. Suppose we are interested in estimating the population total

$$t_j = \sum_{i \in U} x_{ji},$$

given a sample $S = \{1, 2, \ldots, n\} \subset U$ with sampling design weights d_i. An estimate for t_j is the Horvitz-Thompson estimator

$$\hat{t}_j^{HT} = \sum_{i \in S} d_i x_{ji}.$$

Now suppose we know the population total, t_k for some variable k. Then ideally we want that

$$\sum_{i \in S} d_i x_{ki} = t_k,$$

but this is often false. The calibration process is used to find weights w_i for $i \in S$ close to d_i, based on a distance function, such that

$$\sum_{i \in S} w_i x_{ki} = t_k.$$

After finding w_i, the calibration estimator for t_j is

$$\hat{t}_j = \sum_{i \in S} w_i x_{ji}.$$

In section 2.1, we briefly describe the calibration problem; while in section 2.2, we explain the algorithms developed; and in section 2.3, we present a package for calibrating integer weights with

R software (R Core Team, 2015).

2 METHODS

2.1 Description of the problem

Normally, one approaches the calibration problem by solving a system of linear equations. This linear system is formulated as

$$y = Aw,$$

where y denotes the vector of point targets, A represents the matrix of collected data, and w is the vector of weights to calibrate.

Usually, the number of equations is less the number of weights, and so there exists multiple solutions. One solution is given by computing the product between the Moore-Penrose inverse of the matrix A and the vector of point targets. However, this method can be computationally unfeasible due to the large dimensions of the data matrix. Therefore, one usually reduces the problem to the following optimization:

$$\min_{w \in \mathcal{W} \subset \mathbb{R}^n} L(y - Aw), \tag{1}$$

by satisfying the following constraints:

$$Aw \geq \ell_y, \text{ and } Aw \leq u_y,$$

where ℓ_y and u_y are two vectors representing respectively the lower and the upper bounds around the point targets, and $\mathcal{W}$ is the set of integer numbers which satisfy the following inequalities:

$$w \geq \ell_w, \text{ and } w \leq u_w. \tag{2}$$

The objective function $L(\cdot)$ is based on a generic loss function, which can be also used to improve convergence and further simplify the problem.

2.2 Description of the algorithm

Traditionally, the problem in (1) can be attacked by performing the minimization with real numbers (Kott, 2006), and then rounding to the best integer solution. Here, we propose a method to estimate the calibration weights by dealing only with integer numbers. The original problem can be simplified by considering the following formula:

$$\min_{w \in \mathbb{Z}^n} \sum_{i=1}^{n} \rho_{y_i,\ell_i,u_i}\left(a_i^\top w\right) + \lambda P(w)$$

such that the constraints in (2) are satisfied, where a_i is the i-th row of the matrix A, $\rho_{y_i,\ell_i,u_i}(\cdot)$ is a generic loss function, and λ is a non-negative scalar, which balances the weight penalty $P(\cdot)$ which can be defined as $\|w - d\|_1$, where d can be either the adjusted or unadjusted non-integer initial choice of the design weights.

2.2.1 Adjustments and an ad-hoc rounding method

In order to produce a feasible initial point for the calibration algorithm, the non-integer initial choice of weights must be processed. Initially, whenever the constraints in (2) are not satisfied, the all unfeasible weights are truncated to their closest boundary. The gradient of the objective function is then computed from the adjusted weights for assigning a higher priority to the weights to be changed first. Hence, the gradient is calculated by

$$\sum_{i=1}^{n} \nabla_w \rho_{y_i,\ell_i,u_i}\left(a_i^\top w\right) - \lambda \operatorname{sign}(w - d), \tag{3}$$

where $\nabla_w \rho_{y_i,\ell_i,u_i}\left(a_i^\top w\right)$ is the gradient of the chosen loss function, while the function sign$(\cdot)$ is used to compute the sign of each component of the $w - d$ vector. The priority is given from the largest to the lowest absolute value of the gradient components. For each weight to be rounded, either the closest lower or upper integer is selected such that the objective function is minimized. Whenever the objective function is not sufficient to establish which integer number is better, the regular rounding technique is applied. This technique is iterated by conditioning on the rounded weights computed in the previous steps. At each step, the gradient is updated for computing the new priorities. In so doing, the best rounded point is achieved to allow for the execution of the calibration algorithm.

2.2.2 An integer-swapping algorithm for calibration

The integer programming technique, that we called the integer-swapping algorithm, addresses the calibration by dealing with only integer numbers. It is designed to move a singular integer weight by unit-shifts according to the constraints in (2) and the magnitude of the gradient in (3).

The algorithm starts by calculating both the gradient and the priority indexes, which will be updated at each single step, in order to allow for multiple shifts. Successively, the weight with a higher priority index is processed, and since the gradient gives the climbing direction of the objective function, the unit-shifts are established by the opposite sign of the gradient. If the shift decreases the objective function, then the vector of integer weights is updated, and the procedure starts from the beginning; otherwise, the consecutive weight with lower priority is processed as explained above. The algorithm stops when no unit-shift produces any improvement.

2.3 An R package for integer calibration

In this section we present **inca** (Sartore and Toppin, 2015) an R package for the calculation of integer weight calibration. The main feature of this package is the fact that it produces integer weights after calibration that are close to the design weights. There are three primary functions within **inca**. These three primary functions are `adjWeights()`, `roundWeights()` and `intcalibrate()`. We describe each of these functions below.

2.3.1 `adjWeight()`

This function ensures that all weights are within the provided boundaries. If a weight is outside the provided boundaries, `adjWeight()` will change this weight to the closest boundary integer. We illustrate with an example. Let $[0.1, 6.9]$ be provided boundaries. Then the boundary integers are $[1, 6]$. Let 0.7 be a weight in the vector of weights we are calibrating. Then `adjWeight()` will change 0.7 to 1, since 1 is the closest boundary integer. The output from this function is a vector of weights that is within the user defined boundaries.

2.3.2 `roundWeights()`

This function acts like a modified `floor()` or `ceiling()` function. It rounds a weight to either its `floor()` or `ceiling()` depending on which is better for the objective function. The output from this function is a vector or integer weights which is not optimized. The optimization of the weights is the job of the next primary function.

2.3.3 `intcalibrate()`

This function executes the integer-swapping programming algorithm. In **inca**, the user selects from 10 different objective functions listed in Table 1. The `intcalibrate()` function calculates the gradient of the selected objective function and uses this information to determine which weight must be changed for minimizing the objective function. The output from this function is the final vector of the calibrated integer weights.

Objectives	**Description**
`L1`	Sum of absolute errors
`aL1`	Asymmetric sum of absolute errors
`rL1`	Sum of absolute relative errors
`LB1`	Sum of absolute errors of outside boundaries
`rB1`	Sum of absolute relative errors if outside the boundaries
`L2`	Sum of square error
`aL2`	Asymmetric sum of square errors
`rL2`	Sum of square relative errors
`LB2`	Sum of square errors if outside the boundaries
`rB2`	Sum of the square relative errors if outside the boundaries

Table 1: Table showing the different objective functions available in **inca** and their description.

3 RESULTS

A simple simulation is performed to demonstrate the use of the **inca** package. The dataset `A`, the true weights `tw`, and the targets `y` are initially generated. Then 154 weights are simulated as an initial guess from a uniform distribution with support in $[0, 7.77]$. If each weight is assumed to be in the set $\{1, \ldots, 6\}$, then the weights are rounded and calibrated through the following R code:

```
set.seed(0); tw <- rpois(154, 3) + 1

A <- matrix(rbinom(154000, 1, .3) * rpois(154000, 4), 1000, 154)
```

```
y <- A %*% tw; iw <- runif(154, 0, 7.77) # Initial guess
rw <- inca::roundWeights(iw, ~. + 0, y, l = 1, u = 6, sp = T, d = A)
cw <- inca::intcalibrate(rw, ~. + 0, y, l = 1, u = 6, sp = T, d = A)
```

In Figure 1 we see how the real design weights are transformed by calibration.

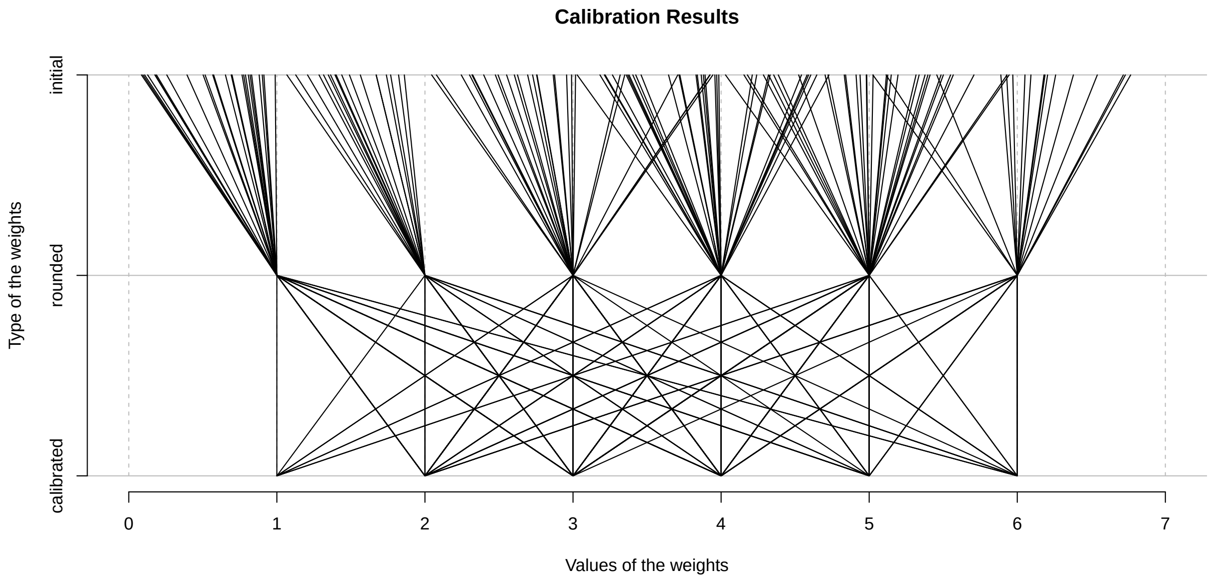

Figure 1: Parallel-coordinates plot showing the transition between the design, initial and calibrated weights.

4 CONCLUSIONS

This new integer calibration algorithm is a significant upgrade over existing methods that produce integer weights. It eliminates the need for rounding after real number calibration. We developed an R package, based on our method, to help other R users with integer calibration problems. The main features of the R package **inca** have been explained and illustrated in the package. The current version (0.0.1) of **inca** should be regarded as a package for the calculation of integer weights in calibration. See **inca** package manual for a complete guide on how to use **inca**.

5 REFERENCES

Deville, J.C. 1988. "Estimation linéaire et redressement sur information auxiliaire d'enquêtes par sondage."

Deville, J.C., and C.E. Särndal. 1992. "Calibration Estimators in Survey Sampling." *Journal of the American Statistical Association* 87:376–382.

Kott, P. 2006. "Using calibration weights to Adjust for nonresponse and coverage errors." *Survey Methodology* 32:133–142.

Lemel, Y. 1976. "Une généralisation de la méthode du quotient pour le redressement des enquêtes par sondage." *Annales de l'inséé* :273–282.

R Core Team. 2015. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Sartore, L., and K. Toppin. 2015. *inca: Integer Calibration*. R package version 0.0.1.