WP. 36
ENGLISH ONLY

UNITED NATIONS STATISTICAL COMMISSION and ECONOMIC COMMISSION FOR EUROPE CONFERENCE OF EUROPEAN STATISTICIANS

EUROPEAN COMMISSION STATISTICAL OFFICE OF THE EUROPEAN COMMUNITIES (EUROSTAT)

Joint UNECE/Eurostat work session on statistical data confidentiality
(Geneva, Switzerland, 9-11 November 2005)

Topic (v): Confidentiality aspects of tabular data, frequency tables, etc.

CONTROLLED ROUNDING IMPLEMENTATION

Supporting Paper

Submitted by the Office for National Statistics, United Kingdom and
the University of La Laguna, Tenerife, Spain [1]

---

[1] Prepared by Christine Bycroft, Andrea Toniolo Staggemeier and Juan-José Salazar-González.

The Controlled Rounding Implementation

Juan-José Salazar-González*, Christine Bycroft**, Andréa Toniolo Staggemeier**

* DEIOC, University of La Laguna, Tenerife, Spain. (jjsalaza@ull.es)

** Office of National Statistics, U.K.

Abstract

Rounding methods are common techniques in many statistical offices to protect disclosive information when publishing data in tabular form. Classical versions of these methods do not consider protection levels while searching patterns with minimum information loss, and therefore typically the so-called auditing phase is required to check the protection of the proposed patterns. This paper presents a mathematical model for the whole problem of finding a protected pattern with minimum loss of information, and describe an algorithm to solve it. The base scheme is a branch-and-bound search. If time enough is allowed, the algorithm stops with an optimal solution. Otherwise, an heuristic approach aims at finding a feasible zero-restricted solution. On complicated or infeasible tables where finding a zero-restricted feasible solution cannot be found in a reasonable time, the algorithm generates a non zero-restricted solution, here referred to as *rapid table*. This paper presents a summary of findings from some computational experiments.

Keywords: Statistical Disclosure Control; Controlled Rounding; Integer Linear Programming

1 Introduction

Statistical agencies collect data to make reliable information available to the public. This information is typically made available in the form of tabular data (i.e., a table), defined by cross-classification of a small number of variables. By law, the agencies are obliged to preserve the confidentiality of data pertaining to individual entities such as persons or businesses. There are several methodologies for achieving this goal. We refer the reader to Willenborg and de Waal [10] for a wider introduction to this area, called *Statistical Disclosure Control*.

In this area, experts typically distinguish two different problems. The *primary problem* concerns the problem of identifying the disclosive data, i.e., the cell values corresponding to private information that cannot be released for particular values. The *secondary problem* (also named the *complementary problem*) consists in applying methods to guarantee some "protection requirements" while minimizing the "loss of information". This paper concerns only the secondary problem.

The most popular methodologies for solving the secondary problem are variants of the well-known Cell Suppression and Rounding methods. These two fundamental methodologies will be described next. Nevertheless, the different methodologies are applied by practitioners without sharing a common understanding of protection, thus making a comparison very difficult even on the same data. Furthermore, in practice, some implementations cannot inherently guarantee the protection requirements and great computational effort must be applied to check the proposed output before publication. This checking is called the *Disclosure Auditing* phase and basically consists in computing lower and upper bounds on the original value for each disclosive cell; in the literature there are several techniques to perform this third phase, including linear and integer programming, the Frechet and Bonferroni bounds, and the Buzzigoli and Giusti's shuttle algorithm (see, e.g., Duncan, Fienberg, Krishnan, Padman and Roehrig [4] for references).

When applying a rounding procedure the experts are given a base number and they are allowed to modify the original value of each cell by rounding it up or down to a near multiple of the base number. An output pattern must be associated with the minimum loss of information, which for this methodology can be considered as the distance between the original and the modified tables. In *Random Rounding*, experts decide to round each cell up or down by considering a probability that depends on its original cell value, where the marginal cell values are rounded independently. Therefore, the Random Rounding produces output tables where the marginal values are not the sum of their internal cells, which is a disadvantage of this approach, see Willenborg and de Waal [10] for more details on Unbiased Random Rounding approach.

Controlled Rounding is an alternative methodology that has not been extensively analyzed in the literature, and the aim of this paper is to add new results to fill this gap. Here, probabilities are not considered and the expert should round up or down all cell values such that all the equations in the table hold in the published table. Even not considering protection level requirements, a Controlled Rounding solution may not exist for a given table (e.g., Causey, Cox and Ernst [2] showed a simple infeasible 3-dimensional instance). The difficulties in finding a feasible solution to the problem is due to the highly combinatorial part of the problem, when deciding which cells to round up or down. This is addressed in the literature as an NP-Hard problem for the complexity theory in optimization. Within the exact methods available we referred the reader to Kelly, Golden and Assad [7] who proposed a branch-and-bound procedure for the case of 3-dimensional tables. On the other hand, metaheuristic methods for finding solutions of this problem on multi-dimensional tables have been proposed by several authors, including Kelly, Golden and Assad [7, 8]. The problem was first introduced by Bacharach [1] in the context of replacing nonintegers by integers in 2-dimensional tabular arrays, and actually it arises in several other applications. It was introduced in a statistical context by Cox and Ernst [3]. In all these articles, the protection requirements are not considered and, therefore, they do not address the real problem arising from the Statistical Disclosure Control application. Indeed, a statistical office applying these incomplete approaches must also solve the auditing problem to check the protection of the output (and repeating the procedure when a protection level requirement is

violated). This paper addresses the complete problem of finding a solution of the Controlled Rounding methodology.

Earlier work on implementation of controlled rounding is presented in Salazar et al. [9]. This paper reports on further developments of controlled rounding with a new heuristic approach to solve larger and more complex tables. The requirements of *Neighbourhood Statistics* provide the main motivation for the work to improve the performance of controlled rounding.

The Neighbourhood Statistics website aims to provide a wide range of social and economic tabular data belonging to small areas while maintaining the confidentiality of the data. Tables with small areas can be very large (half to one million cells with the current smallest geographies, and increasing if outputs move to lower level geographies). Tables can also be complex. Geography and sometimes other variables are hierarchical, there may be 3 or 4 dimensions, and there may be linked tables.

The model and algorithm are designed to be general to deal with these complex hierarchical and linked tables as well as with large tables. This new approach is named Heuristic Controlled Rounding Program (HCRP). It has been implemented in C programming language and linked with τ-ARGUS as one of the available confidentiality tools(see Hundepool [5]).

Section 2 introduces the main concepts of the Statistical Disclosure Control problem in a general context. Section 3 considers the well-known *Controlled Rounding Methodology*, presents an Integer Linear programme model, and describes an algorithm for the exact solution. Section 4 describes the implementation of the heuristic approach (HCRP), and a summary of findings from our computational experiments is given in Section 5. These experiments supported the decisions taken during the process of doing the implementation for the best heuristic approach. We end the paper with our conclusions.

2 Basic Concepts and Notation

A statistical agency is provided with a set of n values a_i for $i \in I := \{1, \ldots, n\}$. Vector $a = [a_i : i \in I]$ is known as a "nominal table" and satisfies a set of m equations $\sum_{i \in I} m_{ji} y_i = b_j$ for $j \in J := \{1, \ldots, m\}$. For convenience of notation the linear system will be denoted by $My = b$, thus $Ma = b$ holds. Each solution y of $My = b$ is called a *congruent table* since it is fully additive and coherent with the structure of the original table.

Statistical tables typically contain disclosive data. We denote the subset of disclosive cells by P. In a general situation, all the disclosive cells in a table must be protected against a set K of *attackers*. The attackers are the intruders or data snoopers that will analyze the final product data and will try to disclose confidential information. They can also be coalitions of respondents who collude and behave as single intruders. The aim of the Statistical Disclosure Control is to reduce the risk of them succeeding. Each attacker knows the published linear system $My = b$ plus extra information that bounds each cell value. For example, the simplest attacker is the so-called *external intruder* knowing only that unknown cell values are, say, nonnegative. Other more accurate attackers know tighter bounds on the cell values,

and they are called *internal attackers*. In general, attacker k is associated with two bounds lb_p and ub_p such that $a_p \in [lb_p \dots ub_p]$ for each cell $p \in I$. The literature on Statistical Disclosure Control (see, for example, Willenborg and de Waal [10]) typically addresses the situation where $|K| = 1$, thus protecting the table against the external intruder with only the knowledge of the linear system and some external bounds. Our implementation also considers a single-attacker protection.

To protect the disclosive cell p containing value a_p in the input table, the statistical office is interested in publishing a table that is congruent with a collection of several different possible ones. The output of a Statistical Disclosure Control is generally called a *pattern*, and it can assume a particular structure depending on the methodology considered.

The congruent tables associated with a pattern must differ so that each attacker analyzing the pattern will not compute the original value of a disclosive cell to within a narrow approximation. For each potential intruder, the idea is to define a protection range for a_p and to demand that the a posteriori protection be such that any value in the range is potentially the correct cell value. To be more precise, by observing the published pattern, attacker k will compute an interval $[\underline{y}_p \dots \overline{y}_p]$ of possible values for each disclosive cell p. The pattern will be considered *valid* to protect cell value a_p against attacker k if the computed interval is "wide enough". To set up the definition of "wide enough" in a precise way, the statistical office gives two input parameters for each disclosive cell with nominal value a_p:

- Upper Protection Level: it is a number UPL_p representing a desired lower bound for $\overline{y}_p - a_p$;
- Lower Protection Level: it is a number LPL_p representing a desired lower bound for $a_p - \underline{y}_p$.

The values of these parameters can be defined by using common-sense rules. In all cases, the protection levels are assumed to be unknown by the attacker. An elementary assumption is that

$$lb_p \leq a_p - LPL_p \leq a_p \leq a_p + UPL_p \leq ub_p$$

for each attacker k and each disclosive cell p. For notational convenience, let us also define

$$lpl_p := a_p - LPL_p, \quad upl_p := a_p + UPL_p, \quad LB_p := a_p - lb_p, \quad UB_p := ub_p - a_p.$$

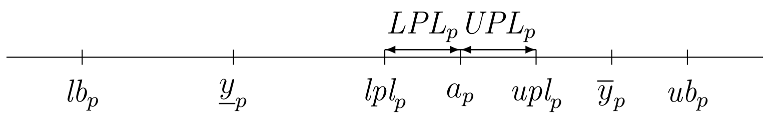

Figure 1: Diagram of parameters

Figure 1 illustrates the position of the parameters in a line. Given a pattern, the mathematical problems of computing values $\underline{y}_p$ and $\overline{y}_p$ are known as *attacker problems* for cell p and attacker k.

An important observation is that we are assuming that our original table contains real numbers. In other words, the value a_i are not necessarily integer. This is a common situation when working with business data. All this documentation is for the general case where a_i are real.

Finally, among all possible valid patterns, the statistical office is interested in finding one with minimum information loss. The *information loss* of a pattern is intended to be a measure of the number of congruent tables in the pattern. A valid pattern must always allow the nominal table to be a feasible congruent table, but it must also contain other different congruent tables so as to keep the risk of disclosure controlled. In practice, since it is not always easy to count the number of congruent tables in a pattern from the point of view of an intruder k, the loss of information of a pattern is replaced by the sum of the loss of information of its cells. In this case, the individual cost for cell p is generally proportional to the difference between the worse-case situations (i.e., to $\overline{y}_p - \underline{y}_p$), it is proportional to the number of respondents contributing to the cell value a_p, or it is simply a positive fixed cost when a_p is not published (i.e., when $\overline{y}_p - \underline{y}_p > 0$).

3 Controlled Rounding Methodology

In *Controlled Rounding Methodology* we are provided with an input base number r_i for each cell i. In practice, the statistical office uses a common base number r_i for all cells, but the method can also be applied when there are different base numbers, as required by some practitioners (e.g., when protecting some hierarchical tables, bigger base numbers are preferred on the top levels than on the low levels). However, in our implementation all base numbers r_i are identical, so from now on all $r_i = r$.

Let us denote by $\lfloor a_i \rfloor$ the multiple of r_i obtained by rounding down a_i, and by $\lceil a_i \rceil$ the multiple of r_i obtained by rounding up a_i. To follow the well-accepted *zero-restricted* version of the Controlled Rounding methodology, if r_i is such that $\lfloor a_i \rfloor = \lceil a_i \rceil$ then we redefine $r_i := 0$, thus $r_i = \lceil a_i \rceil - \lfloor a_i \rfloor$ for all $i \in I$. In other words, cell values which are multiple of the base number are unchanged. See Section 3.3 for more details.

A pattern in the Controlled Rounding methodology is a congruent table $v = [v_i : i \in I]$ such that

$$v_i \in \{\lfloor a_i \rfloor, \lceil a_i \rceil\}. \tag{1}$$

The values r_i are published with the output pattern by the statistical office, thus they are assumed to be known by the attackers. The feasible region for the attacker problems associated with attacker k is defined by

$$\begin{aligned} My &= b & & \\ v_i - r_i \leq y_i &\leq v_i + r_i & \quad & \text{for all } i \in I \\ lb_i \leq y_i &\leq ub_i & \quad & \text{for all } i \in I. \end{aligned}$$

We are not using the general concept of information loss as defined at the end of Section 2. Instead, in controlled rounding the natural concept of "loss of information" of a cell is defined as the difference between the nominal value and the

published value. Then, the loss of information of a pattern is the weighted sum of all the individual loss of information:

$$\delta(v, a) = \sum_{i \in I} w_i |v_i - a_i| \tag{2}$$

We now present a mathematical model for the combinatorial problem of finding a protected Controlled Rounding pattern with minimum loss of information. The optimization problem is referred as *Controlled Rounding Problem* (CRP).

3.1 Preliminary Mathematical Model

Let us consider a binary variable x_i for each cell i, representing

$$x_i = \begin{cases} 0 & \text{if } v_i = \lfloor a_i \rfloor, \\ 1 & \text{if } v_i = \lceil a_i \rceil, \end{cases}$$

i.e., $x_i = 1$ if and only if the published value v_i is obtained by rounding up a_i. Note that when a solution x_i is given, then the published table is determined by $v_i := \lfloor a_i \rfloor + r_i x_i$ for all $i \in I$, and therefore the attacker problems for a given pattern $[x_i : i \in I]$ have a feasible region defined by

$$\left.\begin{array}{rl} My = b & \\ \lfloor a_i \rfloor + r_i x_i - r_i \leq y_i \leq \lfloor a_i \rfloor + r_i x_i + r_i & \quad \text{for all } i \in I \\ lb_i \leq y_i \leq ub_i & \quad \text{for all } i \in I. \end{array}\right\} \tag{3}$$

Notice that, if the original cell values a_i were integers (e.g., frequency tables), then one should add "y_i integer" and replace

$$\lfloor a_i \rfloor + r_i x_i - r_i \leq y_i \leq \lfloor a_i \rfloor + r_i x_i + r_i$$

by

$$\lfloor a_i \rfloor + r_i x_i - (r_i - 1) \leq y_i \leq \lfloor a_i \rfloor + r_i x_i + (r_i - 1).$$

However, remember that all this documentation is for a_i being a real number.

The loss of information of a cell i can now be written as a constant when $x_i = 0$, plus a (positive or negative) parameter c_i if $x_i = 1$. In our implementation c_i represents the cost of rounding up rather than down, then let $c_i = (\lceil a_i \rceil - a_i) - (a_i - \lfloor a_i \rfloor) = \lceil a_i \rceil + \lfloor a_i \rfloor - 2a_i = \lfloor a_i \rfloor + r_i - a_i - (a_i - \lfloor a_i \rfloor) = r_i - 2(a_i - \lfloor a_i \rfloor)$ as in our implementation. Therefore, the loss of information (2) of the pattern v defined by $[x_i : i \in I]$ is a constant plus $\sum_{i \in I} c_i x_i$ when solving the zero-restricted case. Note that c_i are costs defined explicitly for the controlled rounding, and they should not be confused with the weight values used in jj-format files, as done in the τ-ARGUS for cell suppression. The jj-format is described in the manual.

The CRP is to find a value for each x_i such that the total loss of the information in the released pattern is minimized, i.e. the objective function of our problem is:

$$\min \sum_{i \in I} c_i x_i \tag{4}$$

subject to, for each disclosive cell $p \in P$,

$$\max \{y_p : (3) \text{ holds } \} \geq upl_p \quad (5)$$
$$\min \{y_p : (3) \text{ holds } \} \leq lpl_p \quad (6)$$

and

$$x_i \in \{0, 1\} \qquad \text{for all } i \in I. \quad (7)$$

The objective function (4) minimizes the total loss of information c_i for all variables i rounding up (i.e., with $x_i = 1$). The constraint (5) ensures that the attacker will get an upper estimation $\overline{y}_p$ not smaller than the upper protection limit upl_p. Similarly, constraint (6) ensures that the attacker will get a lower estimation $\underline{y}_p$ not larger than the lower protection limit lpl_p. Finally, constraint (7) enforces integer solutions.

Mathematical model (4)–(7) contains all the requirements of the statistical office (in accordance with the definition given in Section 2), and therefore a solution $[x_i^* : i \in I]$ defines an optimal protected Controlled Rounding pattern. The inconvenience is that it is not an easy model to be solved, since it does not belong to the standard (Mixed) Integer Linear Programming (ILP). The first optimization level is defined by the optimization problem minimizing the information loss (4). The second optimization level is composed of the two problems (5)–(6).

It was observed that a drawback of model (4)–(7) is not the number of variables, which is at most the number of cells for both levels. The inconvenience of model (4)–(7) is the fact that there are optimization problems nested in the two levels. A way to avoid this inconvenience is to look for a transformation into a classical ILP model.This has been implemented and the main ideas are detailed below.

3.2 Implemented Mathematical Model

The final mathematical model implemented is the one illustrated in Figure 2. In this model we keep the additivity requirement, and the minimization of the loss of information through the objective function. Additionally, the external bounds and the protection levels are considered on each rounded cell value.

$$\min \sum_{i \in I} c_i x_i$$

subject to:

$$\begin{array}{rl} \sum_{i \in I} m_{ji}(\lfloor a_i \rfloor + r_i x_i) = b_j & \text{for all } j \in J \\ x_i = 1 & \text{if } \lfloor a_i \rfloor < lb_i \text{ or } upl_i > \lceil a_i \rceil \\ x_i = 0 & \text{if } \lceil a_i \rceil > ub_i \text{ or } lpl_i < \lfloor a_i \rfloor \\ x_i \in \{0, 1\} & \text{otherwise.} \end{array}$$

Figure 2: Basic ILP model for Controlled Rounding.

Recall that when a cell value a_i is multiple of the base number $x_i = 0$ and so remains fixed. Note also that the conditions fixing a variable either to 0 or to 1

are required in, for example, a magnitude table with $a_i = 4$, $LPL_i = 2$, $UPL_i = 2$, $lb_i = 0$, $ub_i = +\infty$ and $r_i = 5$. Indeed, these given parameters will try to guarantee a protection interval $[2, 6]$, and therefore all protected zero-restricted patterns must have $x_i = 1$. Exceptionally, these conditions are unnecessary in some special situations. An example is when the table is a frequency table, and the disclosive cells, the external bounds and the protection levels are set in accordance with the criteria proposed in [9]. This is a situation produced when protecting a frequency table with τ-ARGUS. In other words, a cell i with $\lfloor a_i \rfloor < \lceil a_i \rceil$ in a table generated by τ-ARGUS always satisfies

$$lb_i \leq \lfloor a_i \rfloor \leq lpl_i \text{ and } upl_i \leq \lceil a_i \rceil \leq ub_i$$

because $lb_i = \lfloor a_i \rfloor = lpl_i = 0$ and $upl_i = \lceil a_i \rceil < ub_i$. Under these hypothesis the external bounds and the protection levels are useless, and therefore the rounder called by τ-ARGUS will have the "only" task of finding a 0-1 solution satisfying all the $|J|$ equations. We remark the word "only" as still the resolution of the ILP model remains very difficult.

However, for keeping generality in the model (so it will be working also on tables defined with other criteria), the conditions are included in the implementation. Note that these conditions are tested only once when defining the ILP model, and they are as many test as cells. Therefore, doing this checking (even when unnecessary) is small compared with the computational cost of solving the full ILP model.

We did check the protection of the output solution from this model, and it was satisfied in all cases. What is more, even if it looks like a simple mathematical model, finding a feasible solution was a very difficult task on some medium-size tables. The size of the table has a large impact on the complexity of the approach, but also the shape of the table (i.e., the hierarchical and linked internal structure).

3.3 Dealing with infeasibility

The version of the Controlled Rounding methodology modelled in the previous section is known as *zero-restricted*, and can lead to an infeasible optimization problem as observed by Causey, Cox and Ernst [2]. This is due to the strong constraints (1), and this section presents a way of relaxing such conditions to possibly find a congruent table to be released.

When the zero-restricted problem is infeasible, the statistical office is still interested in rounding the cell values and producing a congruent table protected according to the given protection level requirements. To this end, we look for congruent tables where a cell value a_i is allowed to be rounded to a multiple of the base number not necessarily in $\{\lfloor a_i \rfloor, \lceil a_i \rceil\}$. This includes multiples of the base number.

To keep control of the distance of a rounded value v_i from the original value a_i, we solve a sequence of problems where $|v_i - a_i| \leq (s + 1)r_i$ for all $i \in I$ and for a given parameter s. Starting with $s = 0$, the parameter s is incremented by one unit through the sequence. The problem solved at each iteration can be modelled in a similar way as done for the zero-restricted version. We now need two additional integer variables associated with each cell i to keep a linear objective

function penalizing positive values of these variables: a variable x_i^- giving the number of roundings below $\lfloor a_i \rfloor$ and a variable x_i^+ giving the number of roundings over $\lceil a_i \rceil$. These variables can assume values in $\{0, 1, \ldots, s\}$. The output value will be $v_i = \lfloor a_i \rfloor + r_i(x_i + x_i^+ - x_i^-)$. The constraints are the same as for the zero-restricted model but replacing $\lfloor a_i \rfloor + r x_i$ by $\lfloor a_i \rfloor + r_i(x_i + x_i^+ - x_i^-)$. Note that by introducing the new variables, the number of variables in this general model is three times larger than this number in the zero-restricted model. Thus, this will have an impact in the computational time needed to solve the non-zero-restricted model.

In the implementation of the general model, the objective function (4) has been replaced by

$$\min \sum_{i \in I} c_i x_i + r_i x_i^+ + r_i x_i^-$$

Note that in the non zero-restricted model a cell with a value already multiple of the base number will have $x_i = 0$, $x_i^+ \in \{0, 1, \ldots, s\}$ and $x_i^- \in \{0, 1, \ldots, s\}$.

4 Heuristic Approach

Section 3 has outlined the mathematical formulation of the controlled rounding problem, this section will describe how the methods have been implemented in the HCRP, in practical terms. The aim of this section is to produce documentation to explain the general structure of the HCRP.

The implemented algorithm can be summarized as follows. It consists of two methods:

Sophisticated Method : This is a near-optimal approach to find a proper solution for the Controlled Rounding Problem, including the protection requirements. It also tries to find a solution by rounding each cell value to a closer multiple of the base number, up or down. The method basically solves the described mathematical model on figure 2 through a branch-and-bound procedure where the bound is computed by solving a linear-programming relaxation. The current implementation should be observed as a multi-start greedy procedure where, at each node of the branch-and-bound tree, the fractional information is used to build a (potential feasible) integer solution. This is explained in more detail below. The whole method will be referred to here as HCRP. Due to the difficulty of the combinatorial problem, a solution may not exist or, even if a solution exists, the approach could required a very long computational time. Therefore, a potential output of this method is “no solution found”.

Simple Method : This is a fast approach to build a rounded table, called RAPID. RAPID is applied by rounding each internal cell to the nearest multiple of the base number. Then, marginal cell values are obtained by summing the rounded internal cells and the final rounded table is saved on a solution file. The disadvantage of this approach is the quality of the solution, since any of the marginal cells can have rounded values far from the original values. On the other hand, it ensures a solution in case that the HCRP does not find a better one.

The two methods have been combined in a single algorithm, which is the new rounder. The RAPID method is executed while HCRP does not have a feasible integer solution. In this way, even if the user stops the execution of the whole algorithm, a rounded table will be available. This combination of the RAPID method inside the overall HCRP method has been implemented as follows:

1. First, a data structure is built so the sophisticated method can easily go from linear relations to cells and vice-versa. This step is reading the table from disk.

2. Cells with pre-fixed values are identified. Fixed cells are cells with values that are multiples of the base number (including zeros) and any additional cells fixed by the model constraints in Figure 2. This preprocessing is done by the linear-programming solver. This phase can require at most 2 minutes on the large tables used in our experiments. When time is an issue, the first two steps cannot be aborted, and the user is forced to wait until the end.

3. LP relaxation of the model in Figure 2 is built and a branch-and-bound algorithm is started. As a result of this procedure a fractional solution to the problem is built and the search for the integer solution is started by using the "best-bound first":

   (a) Using the fractional solution of the linear-programming relaxation the solver tries to build an integer solution at each node. Parameters for the solver are defined so that variables with fractional values are set to integer one by one. This is called in our paper LOCAL, however for the solver users it can be found as HEURFREQ in XPRESS, for example. When the solution of the linear-programming relaxation is integer then this is a LOCAL solution. The integer LOCAL solution provides an upper bound to the minimization of the information loss. The upper bound value is updated as new LOCAL solution are found. If there is no integer solution branching is required.

   (b) Branching means that we will proceed with a recursive approach where an open subproblem from a list $\mathcal{L}$ is solved and replaced by at most two new open subproblems fixed at either $x_i = 0$ or 1. The first open problem is named *father*, while the subproblems are named *children*. A subproblem is a linear program solved at each node. Each new subproblem is solved before being saved in $\mathcal{L}$, so there is a *lower bound* associated with each subproblem. To solve each subproblem we use a linear-programming solver, like XPRESS or CPLEX.

   (c) We select a cell variable with value closest to 0.5 in the fractional solution since we give priority to variables that are farthest from the desired values of 0 or 1. Internal cells are preferred first than marginal cells, and ties are broken by selecting the cell variable in the largest number of linear equations. Once a variable x_i has been selected, the branching consists in creating one subproblem by adding $x_i = 0$ and another subproblem by adding $x_i = 1$. The variables that have been fixed previously in the father problem (i.e. node) continue fixed in the children.

(d) If the list $\mathcal{L}$ is non-empty (i.e., there is an open subproblem created by a previous branching step), we select one problem from this list with the smallest lower bound, thus ensuring a global lower bound on the optimal solution value. This is a selection criteria called *best-bound first*.

4. A fractional solution derived from the LP relaxation problem (Figure 1) is then passed to the RAPID whenever we do not have an integer solution from HCRP. The fractional values associated with the internal-cell variables in the model are rounded to their closest integer, thus defining a rounded value for each internal cell. These rounded values for the internal cells are used by the RAPID to generate rounded values for the marginal cells. This is a new RAPID solution. However the quality of the obtained rounded table can be poor, although it is a better rounded table than that computed by the RAPID algorithm in the first step. Therefore, the RAPID solution file is upgraded. This is to guarantee that even if the program stops, either by intervention of the user or because of time-limit specifications, an additive and integer solution will be available.

5. The stopping criteria of this algorithm are:

   - the list $\mathcal{L}$ is empty. In this case there are two possibilities. Either this is the best integer solution found, and in this case we say it is optimal. Or no integer solution has been found by LOCAL, and in this case it is infeasible.
   - the time limit inserted by the user is achieved. In this case we do not have optimality proof. In the best case, the LOCAL approach was successfully run and found an integer solution (this is a *feasible solution*). Otherwise, we only have a RAPID solution.
   - The user decides to stop with the first LOCAL solution. Again, this will be a feasible solution, but will not be optimal.

The algorithm described above is for the zero-restricted model. For the general non-zero-restricted mode, the implementation is the same, with the observation that branching can now be done also on x_i^+ and x_i^- variables. Indeed, one can get a fractional solution from a linear-programming relaxation which has all the x_i integers, but still the solution is not feasible if x_i^+ and/or x_i^- contains a non-integer value. In this case the branching phase creates new subproblems by selecting and fixing one of these fractional variables. Since there are more possibilities for each cell i, the search space is larger and therefore the overall resolution may take longer.

5 Computational Experiments

This section points out several findings from computational experiments performed on various tables provided by ONS and others found in the literature.

1. We did not find advantages in strengthening the linear-programming relaxation with Gomory cuts, Benders' cuts, clique inequalities, cover inequalities,... In

principle, these families of inequalities could improve the lower bound computed by solving each linear-programming relaxation, but in our particular optimization problem we did not find this an advantage. The reason could be due to the generality of the matrix M (i.e., the table structure), and therefore spending time looking for violated inequalities does not help to solve the problem.

2. We also conducted experiments where the simplex algorithm was replaced by the Newton Barrier approach, a different method for solving linear programs. Unfortunately, the solutions found by the Newton Barrier were more fractional, and there were numerical difficulties running the Newton Barrier on large tables.

3. We decided to use a LOCAL heuristic based on the solver heuristic because we had to balance a faster overall branch-and-bound scheme, against spending more time on each node. In this way, we allow the algorithm to investigate more nodes of the branch-and-bound tree within the time limit.

4. All the decisions taken were based on our experiments. For example, we decided to run LOCAL at each node of the branch-and-bound tree because we found better results than when using the default value of the solver (every 5 nodes). The reason for that result could be because time is better used exploring more nodes and using the solvers' simple, fast heuristic as much as possible.

5. From empirical results we have observed that HCRP is able to solve instances with up to (about) 100,000 cells, and larger tables should be first broken down by τ-ARGUS into subtables fitting this limitation. This is not to say that HCRP cannot solve larger instances, but it gives an indication of the problem size that can be solved within a reasonable amount of time.

When an input table contains many fixed variables (for example, multiples of the base) the optimization algorithm will only work with the relevant variables, due to the preprocessing phase. At this phase the solver removes the fixed variables and also other more complicated features (e.g., equations which are linearly dependent or others) to reduce the solution space search during the branch-and-bound phases. The time used by this preprocessing is always very small (about 2 minute on a large table) in comparison to the time taken to solve the branch-and-bound phase which can take much longer to solve.

However, we observed that there might be a substantial saving in computational time if an input table enters the rounder without lots of unnecessary variables (e.g., zeros and multiples of the base cell). This saving is thought not to be due to the preprocessing (i.e. fixing the values of unchanged cells to the solver) but because large tables are huge files on hard disk (e.g., 70 Mb for a table with 800,000 cells). Therefore, removing fixed variables can create a much smaller input file. Note that this is also true to the output solution file (i.e., the solutions created by RAPID and LOCAL) required by τ-ARGUS, which must have the same variables as in the input file, so it is not just one huge file but two and they are related in size on hard disk.

	dim	cells	equa	Hierar	S-Type	Time	GAP/MaxDist/Jumps
1	2	102051	2052	No	Optimal	17.34	-/-/-
2	2	204051	4052	No	Optimal	96.00	-/-/-
3	2	408051	8052	No	Optimal	387.32	-/-/-
4	2	459051	9052	No	Optimal	493.60	-/-/-
5	2	510051	10052	No	Optimal	1449.61	-/-/-
6	2	1080051	20052	No	Optimal	2260.24	-/-/-
7	2	655793	125067	Yes	Optimal	1014.96	-/-/-
8	2	437532	83314	Yes	Optimal	504.15	-/-/-
9	2	276675	54691	Yes	Optimal	185.91	-/-/-
10	2	118932	24568	Yes	Optimal	26.97	-/-/-
11	3	148960	58548	No	Rapid	1200.00	-/8/8
12	3	133448	52454	No	Optimal	116.28	-/-/-
13	3	124880	49088	No	Optimal	83.20	-/-/-
14	3	46396	18255	No	Optimal	10.44	-/-/-
15	3	38922	15318	No	Optimal	202.63	-/-/-
16	3	38922	15318	No	Optimal	18.50	-/-/-
17	3	181804	104295	Yes	Rapid	1200.00	-/3777/4349
18	3	121296	69548	Yes	Rapid	1200.00	-/2271/2449
19	3	65296	37860	Yes	Feasible	1200.00	0.0010/-/-
20	3	56616	32490	Yes	Feasible	1200.00	0.0013/-/-
21	3	56616	32490	Yes	Feasible	1200.00	0.0010/-/-
22	3	297388	173447	Yes	Rapid	1200.00	-/31544/1120
23	3	787780	461385	Yes	Rapid	1200.00	-/83283/591

Figure 3: Computational results on 23 real-World instances.

In conclusion, although the combinatorial search will be based on the non-fixed variables, the input and output operations on hard disk will be done on the original set of variables. These operations consume time that on large tables can make a big difference. As a recommendation, it is better to create input tables without unnecessary variables, when possible.

Table 3 shows results from testing a small number of tables of the kind typically produced for NeSS. All tables have one geography variable where there may be several thousand categories for small areas, and one or two other variables. The hierarchy is through the geography variable. All tables had ¡ 1% zero cells. Testing was done on an Intel Pentium 4 machine with 2.8 GHz processor, and 2 GB RAM.

Clear indication of the capability of the HCRP regarding the size of table and its internal structure, as measured by the number of dimensions and hierarchy was observed on our set of experiments. Increasing the table dimension and the table hierarchical aspects impacted directly on the size of the problem that could be solved by HCRP. We found that with the same number of dimensions, and with/without hierarchies that there was a clear increase in the time required for optimal solutions as table size increased.

Other factors such as the table density may also have an effect on the ability to find a solution. However we were not able to identify the impact due to the limited

number of instances we had available for testing.

When looking at the hierarchical data, the 2 dimension tables were always able to find an optimal solution. The 3 dimension tables found good feasible solutions, (i.e. the average gap between upper and lower bounds was less than 0.12%), for tables with less than 2400 categories for the hierarchical geography variable. Only Rapid solutions were found for 3 dimension tables tested with a larger number of hierarchical categories.

From this observation we could draw attention to another threshold for the quality of the rapid solution found from the actual maximum distance between the rounded and original grand total values, and the total number of jumps from the base. These two measures give us an indication of whether the solution is acceptable to users or not.

In the case of the largest non-hierarchical 3 dimension table with 5,320 categories for the geography, the distance and the number of jumps was very small constituting an acceptable solution. However for the hierarchical case, we should avoid trying to solve, within this current time limit of 2400 seconds, tables that had over 10,000 hierarchical categories. Anything above this may produce a solution with huge amount of information loss and many jumps from the rounding base.

6 Conclusions

The zero-restricted model has three times fewer variables than the general model. This means, that we expect the general model to take a longer time to be solved when compared with the zero-restricted model on the same tables. Additionally, the zero-restricted model finds solutions with better quality. Therefore, our first recommendation is to try to solve the zero-restricted model, and to run the general model only when the zero-restricted model is infeasible.

The first RAPID solution, which is computed immediately when HCRP runs, is always poor. The later RAPID solutions (if any) tend to improve the quality, but are still infeasible for the zero-restricted model. In same cases, HCRP proves infeasibility immediately, and therefore the returned RAPID solution is very bad. In these cases, it is worth running the non zero-restricted model, or alternatively increasing the base number.

The first feasible solution found by LOCAL tends to be of sufficiently good quality, thus it makes sense to stop the rounder at that point. It also confirms that it is more important to put emphasis on finding a feasible solution rather than an optimal solution, because if a feasible solution is found then it will also implicitly be a near-optimal solution. This is in contrast to the solution of the Cell Suppression problem where it is easy to find feasible solutions, but these are of poor quality. Finding optimal solutions for cell suppression can be very difficult.

Based on our experiments, if one is interested in finding a feasible solution in a reasonable time (e.g., within 30 minutes), most tables of the order of 100,000 cells with several dimensions and/or hierarchies can be rounded by HCRP. Exceptionally we found a much larger table (simpler structure but not 2-dimensional) was solved easily. In the same way, some very complex but smaller tables may be unsolved even

in several hours of computations.

References

[1] Bacharach, M. (1966) "Matrix Rounding Problem", *Management Science*, **9**, 732–742.

[2] Causey, B.D., Cox, L.H. and Ernst, L.R. (1985) "Applications of Transportation Theory to Statistical Problems", *Journal of the American Statistical Association*, **80**, 903–909.

[3] Cox, L.H. and Ernst, L.R. (1982) "Controlled Rounding", *INFOR*, **20**, 423–432.

[4] Duncan, G. T., Fienberg, S. E., Krishnan, R., Padman, R. and Roehrig, S. F. (2001) "Disclosure Limitation Methods and Information Loss for Tabular Data" in Doyle, P., Lane, J., Theeuwes, J. and Zayatz, L. (editors) *Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies*, Elsevier Science.

[5] Hundepool, A. (2002) "The CASC project", 172–180, in Domingo-Ferrer, J. (editor) *Inference Control in Statistical Databases: From Theory to Practice*, Lecture Notes in Computer Science 2316, Springer.

[6] Jewett, R. (1993) "Disclosure Analysis for the 1992 Economic Census", Working paper, U.S.B.C.

[7] Kelly, J. P., Golden, B. L. and Assad, A. A. (1990) "Using Simulated Annealing to Solve Controlled Rounding Problems", *ORSA Journal on Computing*, **2**, 174–185.

[8] Kelly, J. P., Golden, B. L. and Assad, A. A. (1993) "Large-Scale Controlled Rounding Using TABU Search with Strategic Oscillation", *Annals of Operations Research*, **41**, 69–84.

[9] J.J. Salazar, P. Lowthian, C. Young, G. Merola, S. Bond, D. Brown, "Getting the Best Results in Controlled Rounding with the Least Effort", *Privacy in Statistical Databases* (ed. J. Domingo-Ferrer) LNCS 3050 (2004) 58–72.

[10] Willenborg, L. C. R. J. and de Waal, T. (2001) *Elements of Statistical Disclosure Control.* Lecture Notes in Statistics 155, Springer.