Computational Experience with the Controlled Rounding Problem in Statistical Disclosure Control

Matteo Fischetti, Juan José Salazar, and Alberto Caprara
CPR, Consorzio Padova Ricerche, Padova
Interim Report AT-1/D2

Abstract

Controlled rounding is a widely-used methodology in Statistical Disclosure Control. In this report we give a brief overview of the problem formulation for 2- and 3-dimensional tables. The basic algorithms for these cases have been implemented. Some preliminary computational results are reported.

1 Introduction

We are given an integer *rounding base* β and a vector $[a_i : i \in \mathcal{I}]$ satisfying a certain linear system, say $Ma = b$. For example, vector $[a_i]$ may represent a 2-dimensional table with row and column marginal totals.

The *Controlled Rounding Problem* (CRP) consists of finding a perturbation of every a_i to an integer multiple $\tilde{a}_i$ of β, such that the new vector $[\tilde{a}_i : i \in \mathcal{I}]$ satisfies the linear system $M\tilde{a} = b$, and a certain "distance" between a and $\tilde{a}$ is minimized (or at least bounded by a certain $\epsilon > 0$).

As customary, for any real value z let $\lfloor z \rfloor$ and $\lceil z \rceil$ denote the lower and upper integer part of z, respectively. Moreover, we denote by

$$\lfloor z \rfloor_\beta := \beta \lfloor \frac{z}{\beta} \rfloor$$

and

$$\lceil z \rceil_\beta := \beta \lceil \frac{z}{\beta} \rceil$$

the *lower and upper β-rounding* of z, respectively.

In the sequel we consider the *zero-restricted* version of CRP, in which one requires $|\tilde{a}_i - a_i| < \beta$ for all i, i.e., $\tilde{a}_i \in \{\lfloor a_i \rfloor_\beta, \lceil a_i \rceil_\beta\}$. This implies, in particular, that $\tilde{a}_i = a_i$ whenever a_i is an integer multiple of β.

CRP was first introduced by Bacharach [1] in the context of replacing non-integers by integers in tabular arrays. The problem arises in several application contexts, including statistical disclosure control to protect the privacy of published data; see Willenborg and Waal [9] for details.

Cox and Ernst [3] proved that the zero-restricted CRP associated to any 2-dimensional table with row and column marginal totals is always feasible. They also gave efficient methods for finding controlled roundings optimally close to the original table. These methods are based on the transformation of CRP into a network flow problem, see Section 2.

Causey, Cox and Ernst [2] showed that the zero-restricted CRP on multi-dimensional tables with marginal totals is not always feasible, and gave a simple 2x2x2 counter-example. Kelly, Golden and Assad [8] proposed a branch-and-bound procedure for the case of 3-dimensional tables, based on a linear programming relaxation; see Section 3.

Heuristic methods for CRP on multi-dimensional tables have been proposed by several authors, including Kelly, Golden and Assad [6, 7].

Zero-restricted CPR can be formulated as an Integer Linear Program. To this end, observe that one can with no loss of generality assume that $\beta = 1$ (if this is not the case, just divide $[a_i]$ by β). Now, consider $[\tilde{a}_i]$ as a vector of decision variables, and define the polytope

$$P_{CRP} := \{\tilde{a} : M\tilde{a} = b, \lfloor a \rfloor \leq \tilde{a} \leq \lceil a \rceil\}$$

containing the (possibly fractional) vectors $\tilde{a}$ which satisfy the given linear system along with the lower and upper bounds derived from zero-restrictdness.

An important observation is that P_{CRP} is never empty, in that it contains the "original vector" $[a_i]$.

By construction, there is a 1-1 correspondence between the *integer* points in P_{CRP} and the feasible CPR solutions. Hence CPR translates into the problem of determining an integer point inside P_{CRP}.

A somehow related problem consists of finding a *vertex* of P_{CRP}. If M is totally unimodular, as in the case of 2-dimensional tables, the two problems are in fact equivalent. Even if this is not the case, however, a vertex of P_{CRP} is likely to contain just a few fractional components (never more than the number of rows in M), hence a vertex can be viewed as a good starting point for heuristic algorithms to determine actual integer CRP solutions.

Classical linear programming theory shows that every nonempty polytope always has a vertex. The proof of this basic result is constructive, and applies iteratively the following procedure to convert any given point of P_{CRP} into a vertex.

Given the current point $\tilde{a} \in P_{CRP}$, let

$$F := \{i : \lfloor a_i \rfloor < \tilde{a}_i < \lceil a_i \rceil\}$$

contain the indexes of the fractional components of $\tilde{a}$ (those which are not equal to the prescribed lower or upper bounds). If the columns of the submatrix of M indexed by F are linearly independent, then the current point $\tilde{a}$ is a vertex of P_{CRP}, and we are done.

Otherwise, there exists a nonzero multiplier vector $[\lambda_i : i \in \mathcal{I}]$ such that $\lambda_i = 0$ for all $i \notin F$ and $\sum_{i \in F} \lambda_i M_i = 0$, where M_i denotes the column of M indexed by i. Notice that such a λ can be found efficiently through well-known numerical techniques. But then for every real ϵ we have that

$$M(\tilde{a} + \epsilon\lambda) = M\tilde{a} = b,$$

i.e., the point $\tilde{a} + \epsilon\lambda$ satisfies again the given linear system. In other words, λ gives a

"direction" along which one can perturb the current point without affecting the linear system validity.

Suppose now we start with $\epsilon = 0$, and keep increasing (or decreasing) ϵ until a threshold ϵ^* is reached such that any further increase would lead to a point $\tilde{a} + \epsilon\lambda$ violating a lower or upper bound on the variables. In this situation, one can readily see that the new point $\tilde{a} + \epsilon^*\lambda$ has at least one more integer component than $\tilde{a}$, i.e., the set F associated with the new point has fewer elements. One can then replace $\tilde{a}$ by $\tilde{a} + \epsilon^*\lambda$, and repeat the procedure until the fractional support F of the current point corresponds to a set of linearly independent columns.

2 CPR on 2-dimensional tables

Let us consider a rounding base $\beta = 1$ and a "nominal" 2-dimensional table $[a_{ij} : i = 0, 1, \ldots, n;\ j = 0, 1, \ldots, m]$ of reals satisfying the following system $Ma = 0$ (marginal totals):

$$\sum_{i=1}^{n} a_{ij} - a_{0j} = 0, \qquad \text{for all } j = 0, 1, \ldots, m,$$
$$\sum_{j=1}^{m} a_{ij} - a_{i0} = 0, \qquad \text{for all } i = 0, 1, \ldots, n.$$

The zero-restricted CRP asks for rounding every nominal value a_{ij} to $\tilde{a}_{ij} \in \{\lfloor a_{ij} \rfloor, \lceil a_{ij} \rceil\}$ so as to satisfy:

$$\sum_{i=1}^{n} \tilde{a}_{ij} - \tilde{a}_{0j} = 0, \qquad \text{for all } j = 0, 1, \ldots, m,$$
$$\sum_{j=1}^{m} \tilde{a}_{ij} - \tilde{a}_{i0} = 0, \qquad \text{for all } i = 0, 1, \ldots, n.$$

The new table $\tilde{a}$ is called a *consistent rounding* of the original table a.

As already observed in the introduction, the system matrix M is totally unimodular in this case, hence every vertex of polytope P_{CRP} is integer. In this situation one can then solve CRP efficiently by applying standard linear programming techniques.

Effective algorithms are based on a network-flow interpretation of the above linear system. Consider the following (directed) network $G = (V, A)$ with $|V| = n + m + 2$ nodes. G has a *row node* r_i associated to every row i of M, and a *column node* c_j associated to every column j of M. The graph has the following arcs:

- an arc (r_i, c_j) for every row $i \neq 0$ and every column $j \neq 0$,

	Percentage of Zeros				
$m \times n \times p$	0%	25%	50%	75%	90%
100×100	1.67	1.01	0.59	0.28	0.11
200×200	9.04	6.92	4.51	2.09	0.57
300×300	25.28	18.91	12.69	5.91	1.83

Table 1: Average computing time (over 200 random instances) in PC Pentium/75 seconds

- an arc (c_0, r_i) for every row $i \neq 0$,
- an arc (c_j, r_0) for every column $j \neq 0$,
- an arc (r_0, c_0).

Every arc in the network corresponds to an entry a_{ij} of a table, and has two associated lower and upper capacity bounds equal to $\lfloor a_{ij} \rfloor$ and $\lceil a_{ij} \rceil$, respectively.

By construction, there is a 1-1 correspondence between the consistent roundings of the original table and the *integer flow circulations* in the associated network. It then follows that a consistent rounding minimizing a given cost function can be found efficiently by solving a min-cost flow problem on the network.

We have implemented this idea by using the network simplex algorithm embedded in the commercial package CPLEX 3.0. Computational analysis has been performed on 3,000 random instances solved on a PC Pentium/75 notebook. The objective was finding consistent roundings with minimum distance from the nominal table. The base number was fixed to $\beta = 3$.

Table 1 reports average computing times for several possible "Percentage-of-Zeros" densities (defined as the percentage of table entries whose nominal value is already a multiple of base 3).

When no objective function is given, a simpler computation can be performed to find an integer flow circulation. This is in the spirit of the previously-described procedure to detect vertices of a polytope, as it applies to the network-flow interpretation of the equation system $Ma = 0$.

We consider the initial (feasible and fractional) flow circulation f given by $f_{ij} := a_{ij}$ for all i, j, and apply iteratively the following procedure until all the flow components become integer. Define the *incremental network* $G(f) = (V, A(f))$ associated with the current flow $[f_{ij}]$. For every arc (i, j) in G with $\lfloor a_{ij} \rfloor < f_{ij} < \lceil a_{ij} \rceil$, the incremental network has two arcs with opposite directions, namely a *forward arc* (i, j) and a *backward arc* (j, i). By construction, circuits in $G(f)$ correspond to flow re-routing, i.e., to patterns of linearly

	Percentage of Zeros				
$m \times n \times p$	0%	25%	50%	75%	90%
100×100	0.38	0.26	0.18	0.11	0.08
200×200	3.03	1.91	1.12	0.56	0.34
300×300	10.06	6.18	3.37	1.51	0.81

Table 2: Average computing time (over 200 random instances) in PC Pentium/75 seconds

dependent columns of the system matrix M. Hence any such circuit gives a "perturbation direction" along with one can get a new flow circulation f' with one less fractional flow component.

The above algorithm has been implemented in C and ran on a PC Pentium/75 notebook. Table 2 reports average computing times on the same random instances considered in the previous table.

According to the table, the method finds consistent roundings of 2-dimensional tables with up 300 rows and 300 columns in about 10 seconds.

3 CRP on 3-dimensional tables

We are given the rounding base $\beta = 1$ and a 3-dimensional table $[a_{ijk} : i = 0, 1, \ldots, n; j = 0, 1, \ldots, m; k = 0, 1, \ldots, p]$ of reals satisfying the following system $Ma = 0$:

$$\sum_{i=1}^{n} a_{ijk} - a_{0jk} = 0, \qquad \text{for all } j = 0, 1, \ldots, m, \text{ and for all } k = 0, 1, \ldots, p,$$
$$\sum_{j=1}^{m} a_{ijk} - a_{i0k} = 0, \qquad \text{for all } i = 0, 1, \ldots, n, \text{ and for all } k = 0, 1, \ldots, p,$$
$$\sum_{k=1}^{p} a_{ijk} - a_{ij0} = 0, \qquad \text{for all } i = 0, 1, \ldots, n, \text{ and for all } j = 0, 1, \ldots, m.$$

The zero-restricted CRP asks for rounding every nominal value a_{ijk} to its lower or upper integer part, say $\tilde{a}_{ijk}$, so as to satisfy:

$$\sum_{i=1}^{n} \tilde{a}_{ijk} - \tilde{a}_{0jk} = 0, \qquad \text{for all } j = 0, 1, \ldots, m, \text{ and for all } k = 0, 1, \ldots, p,$$
$$\sum_{j=1}^{m} \tilde{a}_{ijk} - \tilde{a}_{i0k} = 0, \qquad \text{for all } i = 0, 1, \ldots, n, \text{ and for all } k = 0, 1, \ldots, p,$$
$$\sum_{k=1}^{p} \tilde{a}_{ijk} - \tilde{a}_{ij0} = 0, \qquad \text{for all } i = 0, 1, \ldots, n, \text{ and for all } j = 0, 1, \ldots, m.$$

Unlikely the 2-dimensional case, the zero-restricted CRP on 3-dimensional tables can

	Percentage of Zeros				
$m \times n \times p$	0%	25%	50%	75%	90%
$15 \times 2 \times 2$	36	15	18	18	2
$10 \times 3 \times 2$	37	52	45	21	7
$6 \times 5 \times 2$	82	92	81	35	9
$5 \times 4 \times 3$	140	156	129	59	12

Table 3: Number of instances requiring branching (out of 1000 trials).

	Percentage of Zeros				
$m \times n \times p$	0%	25%	50%	75%	90%
$15 \times 2 \times 2$	3.89	3.80	5.00	3.33	3.00
$10 \times 3 \times 2$	4.46	4.12	3.98	4.24	3.57
$6 \times 5 \times 2$	3.98	4.72	4.21	3.91	3.22
$5 \times 4 \times 3$	5.07	5.63	5.16	3.98	3.17

Table 4: Average number of esaminated nodes for the instances requiring branching.

be infeasible, see Causey, Cox and Ernst [2]. Moreover, Kelly, Golden and Assad [5] proved the $\mathcal{NP}$-hardness of the problem.

We have implemented a branch-and-bound procedure based on linear programming relaxation. The objective was to determine consistent roundings with minimum distance from the nominal table.

We evaluated the performances of our branch-and-bound method on random instances generated as in Kelly, Golden, Assad and Baker [8]. We generated and solved 20,000 tables with 60 entries, according to different dimensions and density levels. In particular, 1000 tables were generated for each choice of (m, n, p) in $\{(15, 2, 2), (10, 3, 2), (6, 5, 2), (5, 4, 3)\}$ and for percentage-of-zeros density in $\{0, 25, 50, 75, 90\}$. All tables were rounded using base $\beta = 3$.

Table 3 gives the number of instances (out of 1000 trials) requiring branching. Table 4 gives the average number of esaminated nodes when branching is needed.

The computing time for solving each instance in our test-bed never exceeded 0.5 seconds on a PC Pentium/75. This figure shows the effectiveness of our implementation.

Additional experiments have been performed on larger instances. Table 5 gives average results for tables from 4x4x4 to 8x8x8. Column "count" gives the number of instances requiring branching (out of 1000 trials). Columns "nodes" and "time" give the average number of nodes and the average computing time on a PC Pentium/75, respectively, computed with respect to the cases requiring branching. Again, all problems were solved to optimality within short computing time.

m	n	p	Perc.Zeros	count	nodes	time
4	4	4	0	62	4.03	0.29
4	4	4	25	33	4.94	0.27
4	4	4	50	27	5.67	0.25
4	4	4	75	27	5.07	0.23
4	4	4	90	3	3.67	0.19
6	6	6	0	121	17.79	2.32
6	6	6	25	123	12.77	1.57
6	6	6	50	112	13.11	1.16
6	6	6	75	91	12.05	0.66
6	6	6	90	36	6.72	0.28
7	7	7	0	162	40.54	13.66
7	7	7	25	159	40.85	12.18
7	7	7	50	152	24.49	6.58
7	7	7	75	134	25.81	4.63
7	7	7	90	78	10.82	2.49
8	8	8	0	172	102.09	64.97
8	8	8	25	175	83.42	46.19
8	8	8	50	173	68.98	30.01
8	8	8	75	165	58.78	15.13
8	8	8	90	97	18.69	3.18

Table 5: Statistics on larger instances.

4 Non zero-restricted controlled rounding problem

In order to find a feasible rounding when the zero-restricted is infeasible, we have considered the following mathematical model.

Let $a = [a_i : i \in \mathcal{I}]$ be the nominal data set, satisfying a certain linear system, say $\sum_{i \in I} m_{ij} a_i = b_j$ $(j \in \mathcal{J})$, and let β be the base number. Let us associate a variable x_i to each $i \in \mathcal{I}$ representing an integer such that $x_i \beta$ is a possible rounding for a_i $(i \in \mathcal{I})$. Let us given lb_i and ub_i the lower and upper limits, respectively, for a variable x_i $(i \in \mathcal{I})$. Typically $lb_i = \lfloor a_i/\beta \rfloor$ and $ub_i = \lceil a_i/\beta \rceil$, but depending on the statistical office they can be other values.

The problem of finding a feasible rounding (zero-restricted or not) is the problem of finding a feasible solution to the integer linear system:

$$\begin{aligned} \sum_{i \in \mathcal{I}} m_{ij} x_i = b_j/\beta \qquad & \text{for all } j \in \mathcal{J}, \\ lb_i \leq x_i \leq ub_i \qquad & \text{for all } i \in \mathcal{I}, \\ x_i \text{ integer} \qquad & \text{for all } i \in \mathcal{I}. \end{aligned}$$

When there are several feasible solutions to this systems, then it could be interesting to look for the one minimizing a "linear" distance between the nominal and rounded table. To address this idea, we have added the following objective function to the previous system.

$$minimize \sum_{i \in \mathcal{I}} \delta_i x_i$$

where δ_i is 1 if $a_i - lb_i\beta < ub_i\beta - a_i$ and -1 otherwise. This aim favours rounding down a cell when its nominal value is nearest such a value, and rounding up otherwise.

We have implemented a depth-first-search branch-and-bound scheme based on the LP-relaxation of the model at each node.

5 Acknowledgment

Work supported by the European Union - ESPRIT project SDC on Statistical Disclosure Control.

References

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

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

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

[4] L.H. Cox, "A Constructive Procedure for Unbiased Controlled Rounding", *Journal of the American Statistical Association*, 82 (1987) 520–524.

[5] J.P. Kelly, A.A. Assad, B.L. Golden, "Controlled Rounding of Tabular Data", *Operations Research*, 38 (1990) 760–772.

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

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

[8] J.P. Kelly, B.L. Golden, A.A. Assad, E.K. Baker, "Controlled Rounding of Tabular Data", *Operations Research*, 38 (1990) 760–772.

[9] L. Willenborg, T. Waal, "Statistical Disclosure Control in Practice", *Lecture Notes in Statistics 111*, Springer, 1996.