A Tool for Analyzing and Fixing Infeasible RCTA Instances*

Jordi Castro** and José A. González

Department of Statistics and Operations Research,
Universitat Politècnica de Catalunya,
Jordi Girona 1–3, 08034 Barcelona, Catalonia
jordi.castro@upc.edu,
jose.a.gonzalez@upc.edu
http://www-eio.upc.es/~jcastro

Abstract. Minimum-distance controlled tabular adjustment methods (CTA), and its restricted variants (RCTA), is a recent perturbative approach for tabular data protection. Given a table to be protected, the purpose of RCTA is to find the closest table that guarantees protection levels for the sensitive cells. This is achieved by adding slight adjustments to the remaining cells, possibly excluding a subset of them (usually, the total cells) which preserve their original values. If either protection levels are large, or the bounds for cell deviations are tight, or too many cell values have to be preserved, the resulting mixed integer linear problem may be reported as infeasible. This work describes a tool developed for analyzing infeasible instances. The tool is based on a general elastic programming approach, which considers an artificial problem obtained by relaxing constraints and bounds through the addition of extra elastic variables. The tool allows selecting the subset of constraints and bounds to be relaxed, such that an elastic filter method can be applied for isolating a subset of infeasible table relations, protection levels, and cell bounds. Some computational experiments are reported using real-world instances.

Keywords: statistical disclosure control, controlled tabular adjustment, mixed integer linear programming, infeasibility in optimization, elastic constraints, elastic filter.

1 Introduction

Minimum-distance controlled tabular adjustment methods (CTA) were suggested in [1,9] as an alternative to the difficult cell suppression problem (CSP) [2,11]. In some instances, the quality of CTA solutions has shown to be higher than that of

* Supported by grants MTM2009-08747 of the Spanish Ministry of Science and Innovation, SGR-2009-1122 of the Government of Catalonia, and Eurostat framework contract 22100.2006.002-226.532.

** Corresponding author.

J. Domingo-Ferrer and E. Magkos (Eds.): PSD 2010, LNCS 6344, pp. 17–28, 2010.

solutions provided by CSP [4]. Moreover, quality criteria can be easily added to CTA [6]. A variant of CTA, where only a subset of the cells are allowed to be modified (e.g., total cells), is named restricted controlled tabular adjustment (RCTA).

In 2008, RCTA was included in a solution scheme for the protection of structural business statistics released by Eurostat. The resulting RCTA package [5,10] was developed by the authors under the Eurostat framework contract 22100.2006.002-226.532 in collaboration with Statistics Germany and Statistics Netherlands. This package is linked to two state-of-the-art commercial solvers, CPLEX and Xpress, and it can be used as an stand-alone package or a callable library. It offers the user control about the most instrumental parameters for the mixed integer linear optimization problem (MILP) to be solved. In 2009–2010 this package was extended for the protection of animal production statistics of the European Union, again released by Eurostat. One of these extensions was a tool for analyzing infeasible RCTA instances. Infeasibilities in the MILP optimization model for RCTA may arise by many factors—indeed, by interactions of them: (i) protection levels of sensitive cells may be too large, such that the remaining cells can not accommodate to them (i.e., can not be sufficiently perturbed); (ii) the bounds of non-sensitive cells can be too tight (thus limiting the allowed perturbations); (iii) a particular case of previous point (ii) is when bounds are zero, i.e., some cell values have to be preserved in the released table (usually total cells).

Detecting the cause of infeasibility in MILP optimization is much harder than in LP optimization. Indeed, procedures as, for instance, finding an irreducible infeasible set (IIS) are available in some state-of-the-art solvers only for LP problems, but not for MILP problems (an IIS is a minimal set of constraints and bounds which is infeasible, but it becomes feasible if any constraint or bound is removed). In addition, IIS is in general very time consuming for medium-large instances. For this reason we have considered a general elastic programming approach, which is efficient even for moderately large instances. Briefly, the elastic programming approach computes a minimal relaxation of the constraints and bounds of the problem. The recent monograph [7]—and the many references herein—surveys the state-of-the-art in detecting infeasibility in optimization.

The paper is organized as follows. Section 2 reviews the RCTA MILP formulation. Section 3 shows the methodology underlying the infeasibility repair tool developed. Section 4 describes some of the features of the infeasibility repair tool developed, illustrated by an example. Finally, Section 5 reports computational results with some real-world RCTA instances.

2 The RCTA Problem

Given (i) a set of cells $a_i, i = 1, \ldots, n$, that satisfy some linear relations $Aa = b$ (a being the vector of a_i's); (ii) a lower and upper bound for each cell $i = 1, \ldots, n$, respectively l_{a_i} and u_{a_i}, which are considered to be known by any attacker; (iii) a set $\mathcal{S} = \{i_1, i_2, \ldots, i_s\} \subseteq \{1, \ldots, n\}$ of indices of sensitive cells; (iv) and a lower and upper protection level for each sensitive cell $i \in \mathcal{S}$, respectively lpl_i and upl_i,

such that the released values satisfy either $x_i \geq a_i + upl_i$ or $x_i \leq a_i - lpl_i$; the purpose of CTA is to find the closest safe values $x_i, i = 1, \ldots, n$, according to some distance L, that makes the released table safe. This involves the solution of the following optimization problem:

$$\begin{array}{ll} \min\limits_{x} & ||x - a||_L \\ \text{s. to} & Ax = b \\ & l_{a_i} \leq x_i \leq u_{a_i} \quad i = 1, \ldots, n \\ & x_i \leq a_i - lpl_i \text{ or } x_i \geq a_i + upl_i \quad i \in \mathcal{S}. \end{array} \tag{1}$$

Problem (1) can also be formulated in terms of deviations from the current cell values. Defining $z_i = x_i - a_i, \quad i = 1, \ldots, n$ —and similarly $l_{z_i} = l_{x_i} - a_i$ and $u_{z_i} = u_{x_i} - a_i$—, (1) can be recast as:

$$\begin{array}{ll} \min\limits_{z} & ||z||_L \\ \text{s. to} & Az = 0 \\ & l_{z_i} \leq z_i \leq u_{z_i} \quad i = 1, \ldots, n \\ & z_i \leq -lpl_i \text{ or } z_i \geq upl_i \quad i \in \mathcal{S}, \end{array} \tag{2}$$

$z \in \mathbb{R}^n$ being the vector of deviations. Using the L_1 distance, and after some manipulation, (2) can be written as

$$\begin{array}{ll} \min\limits_{z^+, z^-, y} & \sum\limits_{i=1}^{n} w_i (z_i^+ + z_i^-) \\ \text{s. to} & A(z^+ - z^-) = 0 \\ & 0 \leq z_i^+ \leq \quad u_{z_i} \quad i \notin \mathcal{S} \\ & 0 \leq z_i^- \leq -l_{z_i} \quad i \notin \mathcal{S} \\ & upl_i \; y_i \leq z_i^+ \leq u_{zi} \; y_i \quad i \in \mathcal{S} \\ & lpl_i (1 - y_i) \leq z_i^- \leq -l_{zi} (1 - y_i) \quad i \in \mathcal{S} \\ & y_i \in \{0, 1\} \quad i \in \mathcal{S}, \end{array} \tag{3}$$

$w \in \mathbb{R}^n$ being the vector of cell weights, $z^+ \in \mathbb{R}^n$ and $z^- \in \mathbb{R}^n$ the vector of positive and negative deviations in absolute value, and $y \in \mathbb{R}^s$ being the vector of binary variables associated to protections senses. When $y_i = 1$ the constraints mean $upl_i \leq z_i^+ \leq u_{zi}$ and $z_i^- = 0$, thus the protection sense is "upper"; when $y_i = 0$ we get $z_i^+ = 0$ and $lpl_i \leq z_i^- \leq -l_{z_i}$, thus protection sense is "lower". Model (3) is, in general, a (difficult) MILP.

If the problem has negative protection levels (i.e., $lpl_i < 0$ or $upl_i < 0$ for at least one cell i), the optimization model (3) is no longer valid (let us name it the "classical" model). Problems with negative protection levels can be useful for the sequential protection of correlated tables (indeed, this feature was needed for the protection of real-world data and it was added to the RCTA package in a recent extension [5]). For problems with negative protection levels the following alternative model may be used [3]:

$$\begin{array}{ll} \min\limits_{z^+,z^-,y} & \sum\limits_{i=1}^{n} w_i(z_i^+ + z_i^-) \\ \text{subject to} & A(z^+ - z^-) = 0 \\ & l_z \leq z^+ - z^- \leq u_z \\ & z_i^+ - z_i^- \geq upl_i y_i + l_{z_i}(1 - y_i) \quad i \in \mathcal{S} \\ & z_i^+ - z_i^- \leq -lpl_i(1 - y_i) + u_{z_i} y_i \quad i \in \mathcal{S} \\ & (z^+, z^-) \geq 0 \\ & y_i \in \{0, 1\} \quad i \in \mathcal{S}. \end{array} \tag{4}$$

The main difference between (4) and (3) is that (z^+, z^-) are not related to upper and lower protection deviations in (4), but they are just auxiliary variables to model the L_1 distance. As a result, model (4) is valid for any kind of instance, with either positive or negative protection levels. However, it is less efficient than the classical model (3), and then, for problems with only positive protection levels, the classical model is in general a better option [3].

3 The Elastic Programming Approach for Analyzing Infeasibility

Elastic constraints (and bounds) are constraints (and bounds) that can be relaxed (i.e., violated, stretched) by a certain amount. This amount is represented by one or two artificial variables for each relaxed constraint and bound. The elastic constraints for general inequality and equality constraints are:

Nonelastic constraints	Elastic constraints
$A_1 x \geq b^1$	$A_1 x + s^1 \geq b^1$
$A_2 x \leq b^2$	$A_2 x - s^2 \leq b^2$
$A_3 x = b^3$	$A_3 x + s^3 - s^4 = b^3,$

where all the artificial variables s^1, s^2, s^3, s^4 are nonnegative. Once the artificial variables have been added to all the constraints and bounds (or a subset of them), the elastic problem to be solved is to minimize a function of the artificial variables (according to some objective) subject to the elastic constraints and bounds, and to the remaining constraints and bounds that were not relaxed (if any). Some of the different objectives that can be used are: (1) minimize the sum of artificial variables, i.e., $||s^1||_1 + ||s^2||_1 + ||s^3||_1 + ||s^4||_1$; (2) minimize the Euclidean distance of the artificial variables , i.e., $||s^1||_2^2 + ||s^2||_2^2 + ||s^3||_2^2 + ||s^4||_2^2$; (3) minimize the number of relaxed constraints. Objectives (2) and (3) give rise respectively to a quadratic and a MILP problem, even if the original problem was neither quadratic nor MILP. Objective (1) has been our choice for RCTA.

Applying the above elastic programming approach to the RCTA model (3) the resulting problem is

$$
\begin{aligned}
f^* = \min_{s^i, i=1,\ldots,10} \; & \sum_{i=1}^{10} \sum_{j=1}^{n_i} c_j^i s_j^i \\
\text{s. to } & A(z^+ - z^-) + s^1 - s^2 = 0 \\
& z_i^+ - s_i^3 \leq u_{z_i} && i \notin \mathcal{S} \\
& z_i^+ + s_i^4 \geq 0 && i \notin \mathcal{S} \\
& z_i^- - s_i^5 \leq -l_{z_i} && i \notin \mathcal{S} \\
& z_i^- + s_i^6 \geq 0 && i \notin \mathcal{S} \\
& z_i^+ - upl_i \; y_i + s_i^7 \geq 0 && i \in \mathcal{S} \\
& z_i^+ - u_{zi} \; y_i - s_i^8 \leq 0 && i \in \mathcal{S} \\
& z_i^- + lpl_i \; y_i + s_i^9 \geq lpl_i && i \in \mathcal{S} \\
& z_i^- + l_{zi} \; y_i - s_i^{10} \leq -l_{zi} && i \in \mathcal{S} \\
& y_i \in \{0, 1\} && i \in \mathcal{S} \\
& s^i \geq 0 && i = 1, \ldots, 10,
\end{aligned}
\tag{5}
$$

where c^i denotes a penalty vector for each vector of artificial variables s^i, n_i denotes the dimension of each vector $s^i, c^i, i = 1, \ldots, 10$, and f^* is the optimal objective function value obtained. Similarly, the elastic version of the RCTA model (4) for problems with negative protection levels is

$$
\begin{aligned}
f^* = \min_{s^i, i=1,\ldots,10} \; & \sum_{i=1}^{6} \sum_{j=1}^{n_i} c_j^i s_j^i \\
\text{s. to } & A(z^+ - z^-) + s^1 - s^2 = 0 \\
& z^+ - z^- - s^3 \leq u_z \\
& z^+ - z^- + s^4 \geq l_z \\
& z_i^+ - z_i^- + (l_{z_i} - upl_i) y_i + s_i^5 \geq l_{z_i} && i \in \mathcal{S} \\
& z_i^+ - z_i^- + (-lpl_i - u_{z_i}) y_i - s_i^6 \leq -lpl_i && i \in \mathcal{S} \\
& (z^+, z^-) \geq 0 \\
& y_i \in \{0, 1\} && i \in \mathcal{S} \\
& s^i \geq 0 && i = 1, \ldots, 6.
\end{aligned}
\tag{6}
$$

If all the constraints and variables are relaxed, the solution of either problem (5) or (6) will provide an optimal solution. If only a subset of constraints and bounds are relaxed, then (5) or (6) may still result in an infeasible problem. In this case, the subset of relaxed constraints and bounds should be augmented with some additional constraints and bounds. Once a feasible solution to either (5) or (6) is available, a second optimization problem is solved. The purpose of this second phase is to optimize the objective function in terms of the cell deviations, not the artificial variables, such that the solution provided makes sense for RCTA. In this second phase it is imposed as an additional constraint that the sum of artificial variables is less or equal than f^*, the solution of (5) or (6). We know that this problem is feasible, since at least one solution exists (the one reported by (5) or (6)). Therefore, this second optimization is made up of the objective function of (3) (or (4)), the constraints of (5) or (6), and the extra constraint

$$\sum_{i=1}^{t} \sum_{j=1}^{n_i} c_j^i s_j^i \leq (1 + \delta) f^*,$$

where t is either 10 or 6 (depending on whether we used (5) or (6) in the first phase), and $\delta \geq 0$ is a small parameter (e.g., $\delta = 0.001$) to slightly relax the right-hand-side, thus avoiding infeasibility issues. The second phase may be started from the optimal solution of the first problem.

By selecting and iteratively updating the subset of elastic constraints and bounds it would be possible to isolate the cause of infeasibility, i.e., it could be obtained a subset of constraints such that, if not elasticized, the resulting RCTA instance is infeasible. The elastic filter method [8] is an automatic procedure for generating such a subset of constraints. It iteratively solves a sequence of elastic problems, de-elasticizing at each iteration the constraints with a positive artificial variable in the solution. When the elastic problem becomes infeasible, the set of de-elasticized constraints provides the infeasible subset of constraints (i.e., if they are not relaxed, the resulting RCTA model is infeasible). The main inconvenience of this approach is that for large infeasible RCTA instances, each iteration may take a long execution time. This elastic filter approach has not been implemented in the repair tool described in the next section; however, it can be manually applied by the end-user by providing specific subsets of constraints to be relaxed (as shown below, this is one of the features of the infeasibility repair tool).

4 The Infeasibility Repair Tool

The procedure described in the previous section has been implemented and added to a package for RCTA. The resulting tool is named the "infeasibility repair tool". The repair tool has two different working modes. In the first one, it relaxes all the constraints and bounds. In the second mode, the user may select a subset of table constraints $A(z^+ - z^-) = 0$, constraints imposing protection levels for sensitive cells, and bounds on cells deviations. This information is provided by the user in a file with the format shown if Figure 1.

When a cell (either sensitive or not) is included in the second section of Figure 1, its upper bound is relaxed, but not its lower bound. Note that relaxing lower bounds (which are usually zero in most tables) would provide solutions

```
nr, number of table constraints allowed to be relaxed (may be 0)
table constraint number 1
...
table constraint number nr
nx, number of cells allowed to relax their bounds (may be 0)
cell number 1
...
cell number nx
ns, number of sensitive cells allowed to relax their protection levels (may be 0)
cell number 1
...
cell number ns
```

Fig. 1. Format of file for selecting the subset of constraints and bounds to be relaxed

with negative values, which are meaningless; on the other hand, increasing the lower bound would restrict the problem, instead of relaxing it. Therefore, lower bounds are kept fixed. However, the relaxation could be performed for positive lower bounds. In turn, when a sensitive cell is included in the third section of Figure 1, either the constraints

$$upl_i\ y_i \leq z_i^+ \leq u_{z_i}\ y_i\ \ i \in \mathcal{S}$$
$$lpl_i(1-y_i) \leq z_i^- \leq -l_{z_i}(1-y_i)\ i \in \mathcal{S},$$

of (3) or, if negative protection levels are present, the constraints

$$z_i^+ - z_i^- \geq upl_i y_i + l_{z_i}(1-y_i)\ i \in \mathcal{S}$$
$$z_i^+ - z_i^- \leq -lpl_i(1-y_i) + u_{z_i} y_i\ i \in \mathcal{S}$$

of (4) may be relaxed, which in practice means that both protection levels can be violated. If there exists a solution for the relaxed problem, the tool writes an output file with information about the infeasible cells and infeasible linear table relations.

4.1 Example

The table shown in Figure 2(a), with upper bounds in Figure 2(b), is reported as infeasible by the RCTA package. The file describing this instance in the standard "csplib" format is reported in the Appendix A. If the program is run with default parameters, the following output is obtained:

```
Problem reported as infeasible: optimization terminated
(and not by time limit) with no feasible CTA table
Total CPU time: 0.05
```

If the repair tool is applied, the resulting output is:

```
Repair infeasibility procedure successfully finished, see information file.
Total CPU time: 0.1
```

The information file is:

```
Constraints.
Const. num.     left-hand side  right-hand side
0 infeasibilities detected.

Cells.
0 infeasibilities detected among the variables.

Sensitive cells.
Cell 0 (25.996) under UPL (30)
```

meaning that all the table constraints are satisfied (i.e., the table is additive), all the cells remain between bounds, and there is one sensitive cell that could not fulfill its upper protection level of 30. Note that the value appearing in the

$\mathbf{300}_{\mathbf{40}}^{\mathbf{30}}$	8	5	5	5	$\mathbf{38}_{\mathbf{10}}^{\mathbf{4}}$	361
8	$\mathbf{68}_{\mathbf{10}}^{\mathbf{6}}$	40	76	29	31	252
11	33	20	60	35	44	203
7	28	$\mathbf{36}_{\mathbf{9}}^{\mathbf{3}}$	41	22	63	197
326	137	101	182	91	176	1013

(a)

345	15	10	16	13	44
12	78	46	87	33	36
18	38	23	69	40	51
15	32	41	47	25	72

(b)

326	0	0	5	5	25	361
0	74	40	76	29	33	252
0	35	22	60	35	51	203
0	28	39	41	22	67	197
326	137	101	182	91	176	1013

(c)

Fig. 2. (a) Original infeasible table, with primaries in boldface, lower protections levels as subscripts, and upper protection levels as superscripts; (b) upper bounds for cells, table margins are fixed; (c) Adjusted, nonsafe table after repair infeasibility procedure, with unprotected cell marked in boldface

file refers to the deviation from the initial cell value of 300, so the value for the first cell would be 325.996 (rounded to 326 in Figure 2(c), for convenience), suggesting that if the protection level was 26 instead of 30 the table would have been satisfactorily protected.

If, instead, one is interested in preserving the table linear relations and variable bounds, and only relaxing a subset of the sensitive cells (e.g., sensitive cells with values 38, 68 and 36), the infeasibility repair tool should be fed with the following file:

```
0
0
3
5
8
23
```

Note that the above file matches the format of Figure 1, and that, according to Appendix A, cells 5, 8 and 23 are those with values 38, 68 and 36. Indeed, by removing the cell with value 300 from the file—the one whose protection levels were relaxed in the previous run—we are manually applying the elastic filter method described in Section 3. Running the infeasibility repair tool the following output message is obtained:

```
Repair infeasibility procedure reported relaxed problem is infeasible.
Total CPU time: 0.04
```

It means that only relaxing the protection levels of the three selected cells is not possible to obtain a feasible solution. In this case, the protection levels of

sensitive cell of value 300 have to be relaxed, otherwise the problem becomes infeasible.

5 Computational Results

Most of our available real-world instances—from data provided by Eurostat, and processed by Statistics Germany and Statistics Netherlands—are feasible. Then, for the only purpose of testing, the infeasibility tool was initially applied to a set of feasible real-world instances. We note that the procedure based on elastic programming is equally valid for feasible than for infeasible problems: the only difference is that in the feasible case the sum of elastic variables in the first optimization problem will be zero, and that the solution of the second phase will be a valid RCTA solution. The instances considered are related to structural business statistics, for different NACE sector (C, D and E), and to animal production statistics of the European Union. These instances can be considered difficult, since they have a complex structure. The dimensions of these instances, and the results obtained with the RCTA package, with and without the infeasibility repair tool, are reported in Table 1. Problem names starting with "sbs" and "aps" correspond, respectively, to structural business statistics and animal production statistics instances. Columns n, s and m provide the number of cells, sensitive cells and linear relations of the table. Columns "objective", and "CPU" show the final value of the objective functions, and CPU time in seconds obtained with CPLEX-11, with and without the infeasibility repair tool. For executions with the infeasibility repair tool, the objective function corresponds to the solution of the second optimization problem (when it finished, the objective function of the first optimization—the sum of elastic variables—was zero for feasible instances). The required optimality gap was of at most 5% for all the executions. A time limit of one day of CPU (86400 seconds) was set. All the runs have been performed on a Linux Dell Precision T5400 workstation with 16GB of memory and four Intel Xeon E5440 2.83 GHz processors, without exploitation of parallelism capabilities.

For testing the infeasibility repair tool on a non-trivial infeasible case, the instance sbs-E was modified. Results for the new instance, named sbs-E-infeas, are reported in the last line of Table 1. The problem is reported as infeasible in 0 seconds. If very large upper bounds are considered for cell deviations, then the instance is feasible (with an objective function of 106643) but with two unprotected cells due to numerical issues related to feasiblity tolerances and the large bounds considered. After applying the infeasibility repair tool a (infeasible) solution of objective 89931 in the second optimization problem is reported; the objective of the first optimization problem (i.e., the sum of elastic variables, or sum of infeasibilities) was 709546. In this case, infeasibility is being caused by a single constraint. From Table 1 it is clear that the elastic models are much more computationally expensive than the standard RCTA models. We also mention that CPLEX showed to be more robust than Xpress for the solution of the elastic formulations. In particular, Xpress could not solve any of the "sbs" instances with the repair tool. The smaller "aps" instances could be solved by both solvers.

Table 1. Results with CPLEX-11, with and without applying the infeasiblity repair tool, for some real data of structural business statistics and animal production statistics (provided by Eurostat, and processed by Statistics Netherlands and Statistics Germany)

				Without repair		With repair	
Problem	n	s	m	objective	CPU	objective	CPU
sbs-E	1430	382	991	107955	4	106257	297
sbs-C	4212	1135	2580	314282	52	313888	2225
sbs-D_a	28288	7142	13360	414294 (28.3%)	(1)	(2)	
sbs-D_b	28288	7131	13360	444455 (13.5%)	(1)	(2)	
aps-0102	87	5	35	7.20	0.01	7.20	0.03
aps-0203	87	5	35	67.42	0.01	67.42	0.03
aps-0304	87	5	35	12.07	0.02	12.07	0.02
aps-0405	87	5	35	60.77	0.02	60.77	0.02
sbs-E-infeas	1430	382	991	(3)	0	89931	53

(1) Time limit reached with suboptimal solution (gap in brackets).
(2) Time limit reached with no repair tool solution.
(3) Problem reported as infeasible.

6 Conclusions

Detecting what makes a RCTA instance infeasible may be of great help for data owners. However, isolating the source of infeasibility in a MILP is a difficult task. The tool implemented in this work can be used for obtaining a set of constraints such that, if not relaxed, the instance becomes infeasible. The tool is based on adding extra elastic variables to constraints and bounds. The resulting problem is a MILP one, with a higher number of variables than the original RCTA one, and it requires an efficient MILP solver. Our tool was linked to two of them, CPLEX and Xpress, the former seeming to be the most efficient for the elastic model. The tool may also be used not only for real infeasible instances, but also for problematic instances which are reported as infeasible by numerical tolerances. This tool can be seen as another step towards a reliable RCTA package for tabular data protection.

References

1. Castro, J.: Minimum-distance controlled perturbation methods for large-scale tabular data protection. European Journal of Operational Research 171, 39–52 (2006)
2. Castro, J.: A shortest paths heuristic for statistical disclosure control in positive tables. INFORMS Journal on Computing 19, 520–533 (2007)
3. Castro, J.: Extending controlled tabular adjustment for non-additive tabular data with negative protection levels, Research Report DR 2010-01, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya (2010) (submitted)

4. Castro, J., Giessing, S.: Testing variants of minimum distance controlled tabular adjustment. In: Monographs of Official Statistics, Eurostat-Office for Official Publications of the European Communities, Luxembourg, pp. 333–343 (2006)
5. Castro, J., González, J.A., Baena, D.: User's and programmer's manual of the RCTA package, Technical Report DR 2009-01, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya (2009)
6. Cox, L.H., Kelly, J.P., Patil, R.: Balancing quality and confidentiality for multivariate tabular data. In: Domingo-Ferrer, J., Torra, V. (eds.) PSD 2004. LNCS, vol. 3050, pp. 87–98. Springer, Heidelberg (2004)
7. Chinneck, J.W.: Feasibility and Infeasibility in Optimization: Algorithms and Computational Methods. Springer, Heidelberg (2008)
8. Chinneck, J.W., Dravnieks, E.W.: Locating minimal infeasible constraint sets in linear programs. ORSA Journal on Computing 3, 157–168 (1991)
9. Dandekar, R.A., Cox, L.H.: Synthetic tabular Data: an alternative to complementary cell suppression. Energy Information Administration, U.S. (2002) (manuscript)
10. Giessing, S., Hundepool, A., Castro, J.: Rounding methods for protecting EU-aggregates. In: Eurostat methodologies and working papers. Worksession on statistical data confidentiality, Eurostat-Office for Official Publications of the European Communities, Luxembourg, 255–264 (2009)
11. Kelly, J.P., Golden, B.L., Assad, A.A.: Cell suppression: disclosure protection for sensitive tabular data. Networks 22, 28–55 (1992)

A File of the Example of Figure 2 in "csplib" Format

```
0
34
0 300 1 u 0 345 40 30 0
1 8 1 s 0 15 0 0 0
2 5 1 s 0 10 0 0 0
3 5 1 s 0 16 0 0 0
4 5 1 s 0 13 0 0 0
5 38 1 u 0 44 10 4 0
6 361 1 z 0 0 0 0 0
7 8 1 s 0 12 0 0 0
8 68 1 u 0 78 10 6 0
9 40 1 s 0 46 0 0 0
10 76 1 s 0 87 0 0 0
11 29 1 s 0 33 0 0 0
12 31 1 s 0 36 0 0 0
13 252 1 z 0 0 0 0 0
14 11 1 s 0 18 0 0 0
15 33 1 s 0 38 0 0 0
16 20 1 s 0 23 0 0 0
17 60 1 s 0 69 0 0 0
18 35 1 s 0 40 0 0 0
19 44 1 s 0 51 0 0 0
20 203 1 z 0 0 0 0 0
21 7 1 s 0 15 0 0 0
22 28 1 s 0 32 0 0 0
23 36 1 u 0 41 9 3 0
24 41 1 s 0 47 0 0 0
25 22 1 s 0 25 0 0 0
26 63 1 s 0 72 0 0 0
27 197 1 z 0 0 0 0 0
28 326 1 z 0 0 0 0 0
29 137 1 z 0 0 0 0 0
30 101 1 z 0 0 0 0 0
31 182 1 z 0 0 0 0 0
32 91 1 z 0 0 0 0 0
33 176 1 z 0 0 0 0 0
10
0 7 : 6(-1) 0(1) 1(1) 2(1) 3(1) 4(1) 5(1)
0 7 : 13(-1) 7(1) 8(1) 9(1) 10(1) 11(1) 12(1)
0 7 : 20(-1) 14(1) 15(1) 16(1) 17(1) 18(1) 19(1)
0 7 : 27(-1) 21(1) 22(1) 23(1) 24(1) 25(1) 26(1)
0 5 : 28(-1) 0(1) 7(1) 14(1) 21(1)
0 5 : 29(-1) 1(1) 8(1) 15(1) 22(1)
0 5 : 30(-1) 2(1) 9(1) 16(1) 23(1)
0 5 : 31(-1) 3(1) 10(1) 17(1) 24(1)
0 5 : 32(-1) 4(1) 11(1) 18(1) 25(1)
0 5 : 33(-1) 5(1) 12(1) 19(1) 26(1)
```