Minimum-Distance Controlled Perturbation Methods for Large-Scale Tabular Data Protection

Jordi Castro
Dept. of Statistics and Operations Research
Universitat Politècnica de Catalunya
Pau Gargallo 5, 08028 Barcelona (Catalonia, Spain)
jcastro@eio.upc.es

Technical Report DR 2003-14
June 2003

Report available from http://www-eio.upc.es/~jcastro

Minimum-Distance Controlled Perturbation Methods for Large-Scale Tabular Data Protection

Jordi Castro *

Abstract

National Statistical Agencies routinely release large amounts of tabular information. Prior to dissemination, tabular data needs to be processed to avoid the disclosure of individual confidential information. One widely used class of methods is based on the modification of the table cells values. However, previous approaches were not able to preserve the values of the marginal cells and the additivity relations for a general table of any dimension, size and structure. Moreover, effective methods could only be designed for low-dimensional tables. To fill this void, a unified framework for a new class of controlled perturbation methods is presented. Given a set of tables to be protected, they find alternative ones that, guaranteeing confidentiality, minimize the information loss. This goal is accomplished by computing the minimum-distance values to the original cells that make the released information safe. That means solving a constrained optimization problem, whose variables and constraints are respectively related to the tables cells and additivity relations. In practice, real tables may have millions of cells and thousands of linear relations. Three particular methods from the generic framework are derived and implemented, using the one, two and infinity distances. These three variants are evaluated with the unique standard library for tabular data protection currently available. That library contains both low-dimensional artificially generated problems, and real-world highly-structured ones. Some of the complex instances were contributed by National Statistical Agencies, and, therefore, are good representatives of theirs real needs. Unlike alternative methods, the minimum-distance approach was able to solve all the instances with limited computational resources. Each instance only required few seconds on a standard personal computer. The quality of the solution obtained is studied in detail for seven of the most complex instances. The results show that the minimum-distance framework is an effective and promising approach for the protection of large volumes of tabular data.

Keywords: Statistical Confidentiality, Statistical Disclosure Control, Linear Programming, Quadratic Programming, Interior-Point methods.

1 Introduction

The safe dissemination of data is one of the main concerns of National Statistical Agencies. The released data can be classified as disaggregated or aggregated. Disaggregated data (a.k.a. micro-

*Jordi Castro is associate professor, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, Pau Gargallo 5, 08028 Barcelona, Spain (email: jcastro@eio.upc.es). The work was partially supported by the European Union IST-2000-25069 CASC project. The author thanks Lawrence H. Cox and Ramesh A. Dandekar for helpful comments and suggestions.

		z_1	z_2	
⋮	...	...	...	...
51–55	...	38000$	40000$	...
56–60	...	39000$	42000$	...
⋮	...	...	...	...

(a)

		z_1	z_2	
⋮	...	...	...	...
51–55	...	20	1 or 2	...
56–60	...	30	35	...
⋮	...	...	...	...

(b)

Figure 1: Example of disclosure in tabular data. (a) Average salary per age and zip code. (b) Number of individuals per age and zip code. If there is only one individual in zip code z_2 and age interval 51–55, then any external attacker knows the salary of this single person is 40000$. For two individuals, any of them can deduce the salary of the other, becoming an internal attacker.

data or microfiles) consists of files of records, each record providing the values for a set of variables of an individual. Aggregated data (a.k.a. tabular data) is obtained from microdata crossing two or more variables, which results in sets of tables with a likely large number of cells. It must be guaranteed, for both types of data, that no individual information can be derived from the released information. The available methods for this purpose belong to the field of statistical disclosure control. Good introductions to the state-of-the-art in this field can be found in the monographs Willenborg and de Waal (2000) and Domingo-Ferrer (2002).

In this paper we focus on tabular data protection. Although each cell of the table shows aggregated information for several individuals, there is a risk of disclosing individual data. This is clearly shown in the example of Fig. 1. The table (a) of that Figure gives the average salary for age interval and zip code, while table (b) shows the number of individuals for the same variables. If there was only one individual in zip code z_2 and age interval 51–55, then any external attacker would know the salary of this single person is 40000$. For two individuals, any of them could deduce the salary of the other, becoming an internal attacker. Usually, cells showing information about few individuals are considered sensitive, although other rules can be used in practice. Methods for detecting sensitive cells are out of the scope of this work. A recent discussion about sensitivity rules can be found in Domingo-Ferrer and Torra (2002), and Robertson and Ethier (2002).

Figure 1 showed a two-dimensional example. This can be considered the simplest case. However, in practice we must deal with more complex situations, including multidimensional, hierarchical and linked tables. Multidimensional tables are obtained crossing more than two variables, and they can be individually protected. Hierarchical tables are sets of tables whose variables have a hierarchical relation (e.g., zip code and city). In that case, the total or marginal cells of some tables are internal ones for the others. They have to be protected together, to avoid the disclosure of sensitive data. Finally, linked tables are a generalization of the previous situation, where several tables are made from the same microdata, thus sharing information or cells, either hierarchical or not. Again, they have to be protected together. Linked tables can deal with any table dimension, size and structure, and thus include the other situations. Dealing with linked tables is a desired feature of any tabular protection method. Eventually, the final goal would be the protection of the whole set of linked tables that can be produced from some microfiles (e.g., a population census). Clearly, the number of cells involved in that case might be of several millions, an impractical size for most current tabular protection techniques. The new family of protection methods introduced in this work deal with linked tables, and, as shown in the computational results, can solve real-world large instances in few seconds. All the above situations can both refer to frequency tables (i.e., cell values are integer and are usually associated to the number of individuals in that cell) or magnitude tables (i.e., cell values are real, and, for instance, they show the mean for some other variable of all the individuals in that cell). In this work we focus on tables of magnitudes. For tables of frequencies the procedures here described can also be applied followed by some heuristic post-process.

Current methods for tabular data protection can be classified as perturbative (they change the cell values) or nonperturbative (no change is performed). The most widely used nonperturbative method is *cell suppression*, where some *secondary* cells are removed to avoid the disclosure of some sensitive *primary* cells (which are removed as well). That results in a difficult combinatorial optimization problem, which finds the pattern of secondary suppressions that makes the table safe with a minimum number of cells or information loss. Some heuristics for two and three-dimensional tables (Kelly, Golden and Assad 1992; Carvalho, Dellaert and Osório 1994; Cox 1995; Dellaert and Luijten 1999; Castro 2002) and exact methods for linked tables (Fischetti and Salazar 2000) have been suggested for the cell suppression problem. The main inconvenient of this approach is that, due to its combinatorial nature, the solution of very large instances (with possible millions of cells) can result in impractical execution times.

Among the perturbative approaches, one of the techniques that received more attention was *rounding*. This method rounds cell values to a multiple of a fixed integer rounding base. *Controlled rounding* is a variant where the additivity of the table is preserved (i.e., rounded marginal values are the sum of the corresponding slice of internal rounded cells). Initially introduced in Bacharach (1996), efficient methods could only be developed for two-dimensional tables (Cox and Ernst 1982; Cox 1987), possibly with subtotals (Cox and George 1989). For three-dimensional tables controlled rounding is a NP-hard problem (Kelly, Assad and Golden 1990). Several heuristics (Kelly, Golden and Assad 1990) and exact approaches (Kelly, Golden, Assad and Baker 1990) were devised, but were only applied to small size tables. The NP-hardness of the approach makes it impractical for large tables, as the real-world ones tested in this work. Moreover, in practice it can be necessary to maintain some (possibly all) of the original total cells, instead of rounding them.

To avoid the above lacks of rounding, we suggest a new family of controlled perturbation methods that find the minimum-distance (or closest) tables to those to be protected, preserving marginal values, if required, as well as any set of additional linear constraints. Finding the minimum-distance tables means we try to minimize the information loss when delivering the perturbed values. This approach needs to solve a constrained continuous optimization problem, whose variables and constraints are related to the tables cells and additivity relations, respectively. The formulation of the optimization problem depends on the particular distance used. In this work we examine three of them: L_1, L_2 and L_∞. We'll show that real-world large instances can be efficiently solved using current linear and quadratic programming technology. Independently, Dandekar and Cox (2002) suggested the *controlled tabular adjustment* method. It will be seen that controlled tabular adjustment is equivalent to the minimum-distance approach using the L_1 distance. Recently, Dandekar (2003) introduced an alternative perturbation approach, computationally more efficient that the family of methods here presented. However, such approach can not preserve the value of total cells, which is a desirable property in practice (rather, total cells suffer the largest perturbations). The minimum-distance framework combines both features: is efficient and can preserve total cells.

The structure of the document is as follows. Section 2 introduces the *minimum-distance controlled perturbation* framework. Sections 3, 4 and 5 detail the variants associated to the L_1, L_2

and L_∞ distances, respectively. Section 6 compares the optimization problems derived from those three particular distances. Section 7 analyzes the disclosure risk of the method, showing it is safe. Finally, Section 8 presents some computational results in the solution of some real-world large instances. These computational results are used both to verify the effectiveness of the approach, and to compare the above three distances.

2 The minimum-distance controlled perturbation framework

Any table or list of tables, of any dimension, size and structure, can be represented as an array of cells $a_i, i = 1, \ldots, n$, that satisfy a set of m linear relations

$$\mathbf{M}a = b, \tag{1}$$

$a \in \mathrm{I\!R}^n$ being the vector of a_i's, $b \in \mathrm{I\!R}^m$ the right-hand-side term of the linear relations, and $\mathbf{M} \in \mathrm{I\!R}^{m\times n}$ the cell relations matrix. For instance, for a two-dimensional table of $r+1$ rows and $c+1$ columns (last row and column are marginal) we have

$$\begin{aligned} \sum_{j=1}^{c} a_{ij} &= a_{i,c+1} \qquad i = 1 \ldots r \\ \sum_{i=1}^{r} a_{ij} &= a_{r+1,j} \qquad j = 1 \ldots c. \end{aligned} \tag{2}$$

In the above example, $n = rc$, $m = r + c$, and vector b of (1) would correspond to marginal cell values, implicitly meaning they are fixed. Moving marginal cells to the left-hand-side in (2), we get $n = (r+1)(c+1)$ and $b = 0$, marginal cells thus having the same status—not fixed—that internal cells.

In practice most tables have positive cell values, and constraints

$$a \geq 0 \tag{3}$$

must be added to (1).

Given a set $\mathcal{P}$ of indices of sensitive or confidential cells, the *minimum-distance controlled perturbation* method finds, according to some metric, the closest values x_i to a_i, $i = 1, \ldots, n$, that satisfy the table relations (1) and, if needed, (3), such that $x_i, i \in \mathcal{P}$—the values of the sensitive cells—are safe (safety is discussed below). This model can be applied to any kind of table or set of tables, since it does not constraint the structure of the cell relations $\mathbf{M}a = b$. Any other set of linear relations can also be included to this model. For instance, if needed, we can impose that the values x_i of some cells must be close enough to the original values a_i, i.e., $(1-\alpha)a_i \leq x_i \leq (1+\beta)a_i$, for some small α and β. For cells corresponding to national or regional totals, or for cells with a zero value, $\alpha = \beta = 0$ can be a good choice (i.e., we don't perturb the original cell value). This is usual practice in those situations.

This general model can be formulated as

$$\begin{aligned} \min_x \quad & ||x - a||_L && (4) \\ \text{subject to} \quad & \mathbf{M}x = b && (5) \\ & l_x \leq x \leq u_x, && (6) \end{aligned}$$

$x \in \mathbb{R}^n$ being the vector of perturbed cell values. L in (4) denotes the distance to be used, which can be affected by any positive semidefinite diagonal metric matrix $\mathbf{W} = \text{diag}(w_1, \ldots, w_n)$. In the computational results of Section 8 we used $w_i = 1/a_i$. The three more reasonable choices for L are L_1, L_2 and L_∞. They are discussed in the following sections. (5) guarantees x is a well-formed table. The bounds (6) are used to deal with the level of knowledge any attacker has about the cell values, and to guarantee the safety of the perturbed table, as follows:

- We assume any attacker knows a lower and upper bound, respectively $\underline{a}_i$ and $\overline{a}_i$, for each cell a_i, $i = 1, \ldots, n$. If no previous knowledge is assumed for cell i, we simply set $\underline{a}_i = 0$ ($\underline{a}_i = -\infty$ if bounds (3) were omitted) and $\overline{a}_i = +\infty$. (6) includes bounds $\underline{a}_i \leq x_i \leq \overline{a}_i$.
- The protection of each sensitive cell $i \in \mathcal{P}$ is achieved through a lower and upper protection levels, respectively lpl_i and upl_i, such that the released value should be greater or equal than $a_i + upl_i$ or less or equal than $a_i - lpl_i$. These protection levels are provided by the user (e.g., the National Statistical Agency), and they are usually a fraction of the cell value a_i. We assume that the user fixes in advance the sense of the protection for each sensitive cell. Therefore, (6) includes one of the bounds $x_i \geq a_i + upl_i$ or $x_i \leq a_i + lpl_i$.

If the values of a large number of cells want to be preserved, problem (4–6) can be infeasible. This can happen, e.g., for small instances if marginal cells are maintained in the perturbed table. For large tables, infeasibility will rarely occur. However, if needed, it is possible to modify (4–6) to an alternative formulation as follows. For all cells i that are fixed to the original a_i value, remove bounds $a_i \leq x_i \leq a_i$ in (6), and add the penalization $P||x_i - a_i||_L$ to the objective function (4), P being a large penalty parameter. Due to the large value of P, x_i will be equal to a_i whenever possible. The penalization will intervene in the objective function only if no feasible solution with $xi = a_i$ exists.

If, instead of being a user decision, we want the mathematical programming problem (4–6) to choose the best sense for sensitive cells, either $x_i \geq a_i + upl_i$ or $x_i \leq a_i - lpl_i$, we need a binary variable and two extra constraints for each of them:

$$\begin{array}{rcll} x_i & \geq & -S(1-y_i) + (a_i + upl_i)y_i & i \in \mathcal{P}, \\ x_i & \leq & Sy_i + (a_i - lpl_i)(1-y_i) & i \in \mathcal{P}, \\ y_i & \in & \{0,1\} & i \in \mathcal{P}. \end{array} \qquad (7)$$

S in (7) is a large value (e.g., $S = \sum_{i=1}^n a_i$). When $y_i = 1$, constraints (7) imply $S \geq x_i \geq (a_i + upl_i)$. When $y_i = 0$ we have $-S \leq x_i \leq (a_i - lpl_i)$. That results in a difficult combinatorial optimization problem, which would constraint the effectiveness of the approach to small and medium sized problems. Therefore, instead of solving the combinatorial optimization problem, we can heuristically decide in advance the sense for each sensitive cell ($y_i = 1$ or $y_i = 0$) and then solving the optimization problem (4–6). Some straightforward heuristics were suggested in Dandekar and Cox (2002), but, from the reported computational results, none of them produced significantly better results. The particular choice of y_i values do not affect the safety of the released perturbed table, but only the deviations from the original cell values.

The general problem (4–6) can also be formulated in terms of deviations or perturbations from the current cell values. Indeed, defining

$$x_i = a_i + z_i, \quad i = 1 \ldots n, \qquad (8)$$

the optimization problem (4–6) can be transformed to

$$\min_{z} \quad ||z||_L \tag{9}$$

$$\text{subject to} \quad \mathbf{M}z = 0 \tag{10}$$

$$l_z \le z \le u_z, \tag{11}$$

where $z \in \mathbb{R}^n$ is the vector of deviations, and

$$l_z = l_x - a, \qquad u_z = u_x - a. \tag{12}$$

Two benefits of the formulation in terms of deviations are:

- The cell values a_i of the real table are not needed to solve the optimization problem (9–11). Only the cell relations and deviations bounds, represented by matrix $\mathbf{M}$ and vectors l_z and u_z, are required. Therefore, the solution of the above optimization problem can be performed by an external entity (e.g., if some nonavailable software or hardware was required) without delivering the original cell values.
- Two tables with the same cell relations and bounds, that only differ in the cell values (e.g., corresponding to data of two different years or census), are protected with the same perturbations. Therefore, the optimization problem (9–11) only needs to be solved once.

Next three sections specialize the general model for the L_1, L_2, and L_∞ distances, using the formulation in terms of deviations.

3 The L_1 objective

Using the L_1 distance, the problem (9–11) becomes

$$\begin{array}{ll} \min\limits_{z} & \sum_{i=1}^{n} w_i|z_i| \\ \text{subject to} & (10), (11). \end{array} \tag{13}$$

To transform the above into an equivalent linear programming problem, we replace each z_i by the difference of two nonnegative variables, z_i^+ and z_i^-, associated respectively with the positive and negative deviations:

$$z_i = z_i^+ - z_i^- \ , \quad i = 1, \ldots, n. \tag{14}$$

The resulting linear programming problem is

$$\min_{z^+, z^-} \quad \sum_{i=1}^{n} w_i(z_i^+ + z_i^-) \tag{15}$$

$$\text{subject to} \quad \mathbf{M}(z^+ - z^-) = 0 \tag{16}$$

$$l_z \le z^+ - z^- \le u_z \tag{17}$$

$$z^+ \ge 0, \ z^- \ge 0, \tag{18}$$

$z^+ \in \mathbb{R}^n$ and $z^- \in \mathbb{R}^n$ being respectively the vectors of positive and negative deviations. For cells whose deviations have a zero lower bound, only one of the z_i^+ or z_i^- variables, if any, will have a positive value, since we are minimizing their sum in the objective function. Therefore the term $z_i^+ + z_i^-$ of the objective is equal to $|z_i|$, guaranteeing that problems (13) and (15–18) are equivalent.

Equations (17–18) can be simplified. For a nonsensitive cell i, l_{z_i} and u_{z_i}, as defined in (12), will respectively be negative and positive. Then, for nonsensitive cells, equations (17–18) reduce to

$$\begin{array}{ccccl} 0 & \leq & z_i^+ & \leq & u_{x_i} - a_i \quad i \notin \mathcal{P} \\ 0 & \leq & z_i^- & \leq & a_i - l_{x_i} \quad i \notin \mathcal{P}. \end{array} \tag{19}$$

For a sensitive cell i, the equations to be used depend on the sense of the protection considered, defined in (7) by the binary variable y_i. If the sense is "upper" (i.e., $y_i = 1$) then we must impose

$$\begin{array}{ccccl} upl_i & \leq & z_i^+ & \leq & u_{x_i} - a_i \quad i \in \mathcal{P}, y_i = 1 \\ & & z_i^- & = & 0 \quad\quad\quad\; i \in \mathcal{P}, y_i = 1. \end{array} \tag{20}$$

If the sense is "lower" (i.e., $y_i = 0$) then we need

$$\begin{array}{ccccl} & & z_i^+ & = & 0 \quad\quad\quad\; i \in \mathcal{P}, y_i = 0 \\ lpl_i & \leq & z_i^- & \leq & a_i - l_{x_i} \quad i \in \mathcal{P}, y_i = 0. \end{array} \tag{21}$$

The final linear programming problem to be solved is

$$\min_{z^+, z^-} \; (15) \quad \text{subject to } (16), (19), (20), (21). \tag{22}$$

Using $w_i = 1/a_i$, as in the computational results of Section 8, the objective function to be minimized is the total relative deviation between the original and the perturbed data. (22) is basically the same model of Dandekar and Cox (2002). The only difference is that the formulation in Dandekar and Cox (2002), instead of fixing z_i^- and z_i^+ to 0 in equations (20) and (21), respectively, made them only nonnegative. That can provide wrong results and unsafe tables. For instance, it could happen that, for a cell i with sense "upper protection" (i.e., $y_i = 1$), we had $z_i^+ = z_i^- = upl_i$. That would not violate the constraints imposed in Dandekar and Cox (2002), but the resulting perturbation for that cell, according to (14), is $z_i = upl_i - upl_i = 0$. Therefore the cell would be published unperturbed. The above only applies to sensitive cells, which are forced to have one of the deviations positive. Deviations of nonsensitive cells have zero lower bounds, and a solution with both $z_i^+ > 0$ and $z_i^- > 0$ can never correspond to a minimum.

4 The L_2 objective

Using the L_2 distance, the problem (9–11) becomes

$$\begin{array}{rl} \min\limits_{z} & \sqrt{\sum_{i=1}^{n} w_i z_i^2} \\ \text{subject to} & (10), (11). \end{array}$$

We can remove the square root of the objective, since it does not change the solution point, and makes the optimization problem simpler. The rest of constraints and bounds need not to be modified. In particular, and unlike the L_1 formulation of previous section, negative deviations are not a source of trouble, since they always appear squared in the objective function. The final quadratic optimization problem to be solved is

$$\begin{array}{cl} \min\limits_{z} & \sum\limits_{i=1}^{n} w_i z_i^2 \\ \text{subject to} & (10), (11). \end{array} \tag{23}$$

Using $w_i = 1/a_i$, as in the computational results of Section 8, the objective function corresponds to the χ^2 distance between the original and the perturbed data (L.H. Cox, personal communication, March 26, 2003).

5 The L_∞ objective

In this case, the problem (9–11) is

$$\begin{array}{cl} \min\limits_{z} & \max\limits_{i=1\ldots n} \{w_i |z_i|\} \\ \text{subject to} & (10), (11). \end{array}$$

To remove absolute values, we proceed as in Section 3, replacing each variable by the difference of two positive variables. Moreover, it seems reasonable to consider separately the deviations for the sensitive and nonsensitive cells, since the former are forced to be greater than zero whereas the latter should be as close as possible to zero. The problem to be solved is thus

$$\begin{array}{cl} \min\limits_{z^+, z^-} & \left(\max\limits_{i \in \mathcal{P}} \{w_i(z_i^+ + z_i^-)\} + \right. \\ & \left. \quad \max\limits_{i \notin \mathcal{P}} \{w_i(z_i^+ + z_i^-)\} \right) \\ \text{subject to} & (16), (19), (20), (21). \end{array}$$

To transform the above into a linear programming problem we add two extra variables, $z_{\in \mathcal{P}}$ and $z_{\notin \mathcal{P}}$, which will store the maximum deviation for, respectively, the sensitive and nonsensitive cells. The equivalent linear programming problem can be written as

$$\begin{array}{cl} \min\limits_{z^+, z^-, z_{\in \mathcal{P}}, z_{\notin \mathcal{P}}} & z_{\in \mathcal{P}} + z_{\notin \mathcal{P}} \\ \text{subject to} & (16), (19), (20), (21) \\ & z_{\in \mathcal{P}} \geq w_i(z_i^+ + z_i^-) \quad i \in \mathcal{P} \\ & z_{\notin \mathcal{P}} \geq w_i(z_i^+ + z_i^-) \quad i \notin \mathcal{P}. \end{array} \tag{24}$$

Since (24) is a minimization problem, last two sets of equations force $z_{\in \mathcal{P}}$ and $z_{\in \mathcal{P}}$ to be exactly the maximum (weighted) deviations for each group of cells.

Table 1: Properties of the three optimization problems

	L_1, problem (22)	L_2, problem (23)	L_∞, problem (24)
Number of variables	$2n$	n	$2n+2$
Number of constraints	m	m	$m+n$
Type of problem	linear	quadratic	linear
Solution algorithms	simplex and interior-point	interior-point	simplex and interior-point

6 Comparison of the three optimization problems

The distances of Sections 3–5 gave rise to three different optimization problems, whose main features are shown in Table 1. Only the most efficient solution algorithms for the type of problem are reported. The L_2 objective provides the smallest problem, but it can only be efficiently solved by an interior-point algorithm (Wright 1997). For the other two problems we can either use an interior-point algorithm or the simplex method (Dantzig 1963). The efficiency of those methods depends on the particular structure of the problem (Bixby 2002), and, as it will shown in Section 8, it is difficult to know in advance which will be the fastest option for a particular instance. A theoretical advantage of interior-point algorithms is that they have a polynomial complexity, both for linear and quadratic optimization problems. On the other hand, although the simplex method is nonpolynomial, in practice it is known to be very efficient. It is worth to note that the computational cost for the quadratic problem (23), solved through an interior-point algorithm, is the same as if it was linear, because it has a separable objective function (i.e., there are no products of two different variables) (Wright 1997). Moreover, in the tabular data protection context, interior-point algorithms can be specialized to efficiently solve large instances (Castro 2003).

7 Analysis of the disclosure risk of the method

To retrieve the original cell values a_i from the released ones x_i, an attacker needs the applied deviations z_i. Those deviations are the solution of the optimization problem (9–11). Detailing the expression for the bounds (11), the attacker should then solve

$$\begin{aligned} \min_{z} \quad & ||z||_L & (25)\\ \text{subject to} \quad & \mathbf{M}z = 0 & (26)\\ & z_i \geq \underline{a_i} - a_i, \quad i = 1, \ldots, n & (27)\\ & z_i \leq \overline{a_i} - a_i, \quad i = 1, \ldots, n & (28)\\ & z_i \leq -lpl_i \quad \text{or} \quad z_i \geq upl_i, \quad i \in \mathcal{P}. & (29) \end{aligned}$$

The information required for the solution of (25–29) is:

- The particular distance L used in (25) to compute the deviations. Without this information the attacker should try to solve the problems for L_1, L_2 and L_∞, considering that one of the three solutions gives the required deviations.

- The weights w_i, $i = 1, \dots, n$ used in (25). If $w_i = 1/a_i$, the weights are clearly unknown to the attacker.
- The constraints matrix $\mathbf{M}$ of (26). The attacker knows it from the cell relations of the released table.
- The lower and upper bounds $\underline{a_i} - a_i$ and $\overline{a_i} - a_i$, $i = 1, \dots, n$, of (27) and (28), respectively. $\underline{a_i}$ and $\overline{a_i}$ are the cell value bounds that were assumed known by the attacker when protecting the original table. It can be a strong assumption to consider the attacker knows those exact values. Moreover, the original cell values a_i are clearly unknown to the attacker. However, to correctly solve (25–29) the attacker only needs the same values for the *active* bounds. Active bounds are those satisfied as equalities in the solution. For nonactive bounds it is enough to use values that provide a feasible region larger or equal than for the original problem. For instance, if the attacker guesses that all the bounds resulted inactive when protecting the table, constraints (27) and (28) can be removed. That would be the case if large bounds $\underline{a_i}$ and $\overline{a_i}$ are used by default when protecting tables (e.g., $\underline{a_i} = 0$ and $\overline{a_i} = +\infty$).
- The set $\mathcal{P}$ of sensitive cells of (29). Unlike other protection methods—as cell suppression—, the released table gives no information about which cells are sensitive, or candidates to be sensitive. Therefore, the attacker is forced to deduce sensitive cells from his/her own knowledge.
- The lower and upper protection levels lpl_i and upl_i, $i \in \mathcal{P}$, and the sense ("upper" or "lower") used in (29) for each sensitive cell when protecting the original table. In practice, that information will not be distributed with the released table. Protection levels are usually a percentage of the cell values a_i, which are unknown to the attacker. The number of variations for the protection senses is $2^{|\mathcal{P}|}$. If the senses were, for instance, randomly chosen, the attacker would be unable to reproduce them.

Except for the constraints matrix $\mathbf{M}$, the rest of required terms are unknown or uncertain to the attacker. Therefore, problem (25–29) can not be solved, and the released table will be safe. However, we will analyze two unfavorable situations, where the attacker has respectively partial and complete information about the problem. Although fairly improbable in practice, they are considered to stress the low disclosure risk of the method.

7.1 Attacker with partial information

First, consider the attacker knows L, w_i, that bounds (27) and (28) are inactive—thus can be removed—, the set $\mathcal{P}$ of sensitive cells, and the sense ("upper" or "lower") of each sensitive cell. Without loss of generality, and to simplify the exposition, assume all the senses are "upper". With that information, the safety of the deviations relies on the protection levels upl_i of the sensitive cells. If the attacker can obtain approximate values $upl'_i = upl_i + e_i$, $e_i \in \mathbb{R}$, $i \in \mathcal{P}$, the problem to be solved to disclose the deviations is

$$\begin{array}{ll} \min\limits_{z'} & ||z'||_L \\ \text{subject to} & \mathbf{M}z' = 0 \\ & z'_i \geq upl_i + e_i, \quad i \in \mathcal{P}. \end{array} \tag{30}$$

If $e_i = 0$ for all $i \in \mathcal{P}$, the solution of (30) can provide the deviations used to protect the table. The safety of the table thus depends on how sensitive the solution ${z'}^*$ is to possible small e_i values. The relation between both terms is given by the next proposition, which makes use of the *Lagrange multipliers*, or *dual variables*, of the inequality constraints of (30):

Proposition 1 *If ${z'}^*(e) \in \mathbb{R}^n$ is the solution of (30) for a particular vector of $e = (e_1, \ldots, e_{|\mathcal{P}|})$ values, and $\mu \in \mathbb{R}^{|\mathcal{P}|}$ is the Lagrange multipliers vector of the inequality constraints of (30) for $e = 0$ (i.e., the multipliers obtained when protecting the table), then*

$$\nabla_e ||{z'}^*(e)||_L \Big|_{e=0} = \mu. \tag{31}$$

Proof. This is an immediate result of the *sensitivity theorem* of optimization, which states that, given a problem $\min_x f(x)$ subject to $g(x) \geq d$, and a point $(x^*(d), \mu^*)$, $\mu^* \geq 0$ being the Lagrange multipliers at the solution $x^*(0)$, then $\nabla_d f(x^*(d))|_{d=0} = \mu$. See, e.g., Luenberger (1989, pp. 312–318).

Although not made explicit, the above proposition applies to (30) once formulated as one of the optimization problems (22), (23) or (24). In (22) and (24) the variables were z^+ and z^-. In that case, since we are assuming an upper sense for all the sensitive cells, only the Lagrange multipliers of the bounds $z_i^+ \geq upl_i$ should be considered. Moreover, for, respectively, the L_1 and L_∞ distances, problems (22) and (24) were linear, and the relation (31) can be recast as

$$||{z'}^*(e)||_L - ||z^*||_L = \sum_{i \in \mathcal{P}} \mu_i e_i, \tag{32}$$

z^* being the deviations used to protect the table. Equation (32) holds for small enough vectors $e = (e_1, \ldots, e_{|\mathcal{P}|})$, which are problem dependent. For instance, if (30) is solved through the simplex algorithm, (32) is guaranteed for those vectors $e = (e_1, \ldots, e_{|\mathcal{P}|})$ such that ${z'}^*(e)$ and z^* have the same partition of basic and nonbasic variables (see, e.g., Luenberger (1989, pp. 95–96) for a comprehensive explanation).

If the attacker does not know the set $\mathcal{P}$ of sensitive cells, and uses and approximate one $\mathcal{P}'$, the multipliers of cells $i \in \mathcal{P}' \setminus \mathcal{P}$ will also intervene in (31), decreasing even more the disclosure risk. Proposition 1 gives an indicator of the quality of the protection: tables with non-small Lagrange multipliers for the bounds of deviations are unlikely to be disclosed, even if the attacker has a good knowledge about the original data.

To illustrate the above discussion, consider the example of Figure 2. Table (a) shows the original data to be protected. Sensitive cells appear in boldface, and their upper protection levels upl_i are given in brackets. Using the L_1 distance, weights $w_i = 1$, and bounds $\underline{a_i} = 0$ and $\overline{a_i} = \infty$ for all the internal cells, the optimal deviations computed are shown in Table (b). The objective function value is $||z||_{L_1} = \sum_{i=1}^n |z_i| = 36$. The Lagrange multipliers of the constraints $z_i \geq upl_i$ for the sensitive cells are $\mu_{11} = 0$, $\mu_{23} = 2$, $\mu_{33} = 4$ and $\mu_{34} = 4$. Since bounds (27) and (28) are inactive in the solution, the attacker can use (30) to disclose the deviations of Table (b). If, for instance, the attacker can adjust all the original upl_i protection levels, but for cell a_{11}, (in this case, if $e_{11} \leq 4$, $e_{23} = e_{33} = e_{34} = 0$), from (32) and since $\mu_{11} = 0$, a solution with the same objective function (and possibly with the same deviations) that for Table (b) (i.e., 36) will be obtained. However, if all the protection levels are adjusted with errors, a different solution will be computed. For instance, if problem (30) is solved with $e_{11} = 1$, $e_{23} = 2$, $e_{33} = 3$, $e_{34} = 4$, the

a

$\mathbf{10}_{(\mathbf{3})}$	15	11	9	45
8	10	$\mathbf{12}_{(\mathbf{4})}$	15	45
10	12	$\mathbf{11}_{(\mathbf{2})}$	$\mathbf{13}_{(\mathbf{5})}$	46
28	37	34	37	136

(a)

z

7	0	-6	-1	0
0	0	4	-4	0
-7	0	2	5	0
0	0	0	0	0

(b)

z'

10	4	-11	-3	0
0	0	6	-6	0
-10	-4	5	9	0
0	0	0	0	0

(c)

Figure 2: Example of sensitivity of the method to changes in the protection levels. (a) Original data a to be protected. Sensitive cells are in boldface, and upper protection levels are given in brackets. (b) Optimal deviations z computed with the L_1 distance, weights $w_i = 1$, and inactive bounds $\underline{a_i} = 0$ and $\overline{a_i} = \infty$ for all the internal cells. Marginal cells were fixed. The Lagrange multipliers of the bounds $z_i \geq upl_i$ for the sensitive cells are $\mu_{11} = 0$, $\mu_{23} = 2$, $\mu_{33} = 4$ and $\mu_{34} = 4$. The objective function—the sum of deviations in absolute value—is 36. (c) Deviations z' computed by the attacker using approximate protection levels with errors $e_{11} = 1$, $e_{23} = 2$, $e_{33} = 3$, $e_{34} = 4$. The objective function is 68, which satisfies (32).

deviations z' obtained are those of Table (c). The objective function (i.e., sum of deviations) is 68, which satisfies (32): $68 - 36 = \mu_{11}e_{11} + \mu_{23}e_{23} + \mu_{33}e_{33} + \mu_{34}e_{34}$.

7.2 Attacker with complete information

The attacker may not be able to reproduce the right perturbations through (25–29) even with complete information:

Proposition 2 *Assume the attacker knows all the terms of problem (25–29). If the L_2 distance is used, the solution of that problem will provide the deviations used to protect the table. However, for L_1 or L_2, the attacker can obtain alternative deviations.*

Proof. The objective function of (23), for the L_2 distance, is strictly convex, and thus has a unique minimizer on the feasible region. Therefore, independently of the solution algorithm or implementation used, the attacker will obtain the deviations used to protect the table. For L_1 and L_∞, the objective functions of (22) and (24) are linear, and thus convex, instead of strictly convex. Linear objective functions may have a possible infinite number of minimizers, and different algorithms or implementations can provide alternative solutions.

For instance, Tables (a) and (b) of Figure 3 show two alternative solutions with the L_1 distance for the data of Table (a) of Figure 2. They were obtained with two different implementations of the simplex algorithm, using weights $w_i = 1$, and bounds $\underline{a_i} = 0$ and $\overline{a_i} = \infty$ for all the internal cells. Marginal cells were fixed. The sum of deviations is 36 in both solutions. Table (c) of Figure 3 shows, for the same data, the unique solution for the L_2 distance. Since L_2 involves a quadratic function, the solution attempts to distribute the deviations among all the cells, obtaining a non-integer solution (valid for magnitude tables). The behaviour of the three distances is studied in detail in the next section.

$z^1_{L_1}$

7	0	-6	-1	0
0	0	4	-4	0
-7	0	2	5	0
0	0	0	0	0

(a)

$z^2_{L_1}$

6	0	-6	0	0
1	0	4	-5	0
-7	0	2	5	0
0	0	0	0	0

(b)

z_{L_2}

$3.41\hat{6}$	$3.41\hat{6}$	-6	$-8.\hat{3}$	0
$0.08\hat{3}$	$0.08\hat{3}$	4.0	$-4.1\hat{6}$	0
-3.5	-3.5	2	5	0
0	0	0	0	0

(c)

Figure 3: Example of alternative solutions with complete information by the attacker. The original data a to be protected is that of Table (a) of Figure 2. Sensitive cells are in boldface, and upper protection levels are given in brackets. (a) and (b) Alternative solutions $z^1_{L_1}$ and $z^2_{L_1}$, computed with two different linear programming solvers, using the L_1 distance, weights $w_i = 1$, and bounds $\underline{a_i} = 0$ and $\overline{a_i} = \infty$ for all the internal cells. Marginal cells were fixed. The objective function—the sum of deviations in absolute value—of both solutions is 36. (c) Unique solution z_{L_2} for the L_2 distance, again with weights $w_i = 1$, and bounds $\underline{a_i} = 0$ and $\overline{a_i} = \infty$ for all the internal cells. The two-norm of the deviations vector is 12.12.

8 Computational evaluation

We implemented the three models described in Sections 3–5 using the AMPL modelling language (Fourer, Gay and Kernighan 1993) and CPLEX 8.0 (ILOG CPLEX 2002). We applied them to the CSPLIB test suite, the unique currently available set of instances for tabular data protection (Fischetti and Salazar 2000). CSPLIB can be freely obtained from `http://webpages.ull.es/-users/casc/#CSPlib:`. Although these instances were originally produced for the cell suppression problem, the information provided is the same that for the minimum-distance approach. CSPLIB contains both low-dimensional artificially generated problems, and real-world highly-structured ones. Some of the complex instances were contributed by National Statistical Agencies—as, e.g., Centraal Bureau voor de Statistiek (Netherlands), Energy Information Administration of the Department of Energy (U.S.), Office for National Statistics (United Kingdom) and Statistisches Bumdesant (Germany)—, and therefore are good representatives of theirs real needs. In all the executions a value of at least $a_i + upl_i$ for all $i \in \mathcal{P}$ was imposed (i.e., sense "upper protection" was considered for the sensitive cells), and cell values were weighted by $w_i = 1/a_i$ in the objective function. All runs were carried on a notebook with a Pentium Mobile 4 at 1.8 GHz and 512 Mb of RAM.

We did two groups of computational experiments, which are shown in next two Subsections. In the first group we performed a detailed comparison of the three distances using a small subset of instances. In the second group we solved the remaining CSPLIB instances, again with the three distances.

8.1 Comparing the three distances

For the computational comparison we used the seven most complex instances of CSPLIB, which were also the choice in Dandekar (2003). Those instances are challenging for other approaches, as cell suppression, whereas, as shown below, they can be solved in few seconds with the minimum-distance approach. Table 2 provides their main features: identifier (column "Name"), number of

Table 2: Properties of the seven most complex CSPLIB test instances

Name	Dimensions	Size	n	$\|\mathcal{P}\|$	m
hier13	3D, hierarchical	13,13,13	2020	112	3313
hier16	3D, hierarchical	16,16,16	3564	224	5484
bts4	4D, hierarchical	54,54,4,4	36570	2260	36310
nine5d	9D, linked	4,29,3,4,5,6,5,4,5	10733	1661	17295
ninenew	9D, linked	10,6,6,6,6,6,6,6,6	6546	858	7340
two5in6	6D, linked	6,4,16,4,4,4	5681	720	9629
nine12	9D, linked	10,6,6,6,6,6,6,6,6	10399	1178	11362

dimensions and structure—linked or hierarchical— (column "Dimensions"), size for each dimension (column "Size"), number of total cells and sensitive cells (columns "n" and "$|\mathcal{P}|$", respectively), and number of constraints (column "m"). The structure and size information was obtained from Dandekar (2003).

Tables 3–9 show the results obtained for each instance with the three objective functions. For the L_2 objective function we used the primal-dual interior-point algorithm, which can be considered the most efficient choice. The L_1 and L_∞ objective functions were solved with the two best linear programming algorithms: the simplex method and the primal-dual interior-point method. Although the optimal objective function provided by both algorithms is the same, the solution point returned can be different. For L_1, both the simplex and interior-point solutions were very similar, since all the cells intervene in the objective function. For L_∞, which only considers the sensitive and nonsensitive cells with the maximum deviation, the values obtained for the other cells with the simplex method were much better. In Tables 3–9 we report the results obtained with the simplex solutions for L_1 and L_∞.

For each of the three objective functions, Tables 3–9 show the following information. Row "CPU" gives the CPU time in seconds for each algorithm, simplex or interior-point. Rows "Abs. dev." provide the mean (columns "mean"), standard deviation (columns "std") and maximum (columns "max.") of the absolute deviations (i.e., $|z_i|$) of the cell values, for all the cells (row "all"), for the sensitive cells (row "$\in \mathcal{P}$"), and for the nonsensitive cells (row "$\notin \mathcal{P}$"). A similar information is provided for the percentage absolute deviations (i.e., $100|z_i|/a_i$) in rows "Perc. dev.". Rows "Distr. abs. dev." and "Distr. perc. dev." show, respectively, the distribution of the absolute and percentage deviations, i.e., the number of sensitive and nonsensitive cells (columns "$\in \mathcal{P}$" and "$\notin \mathcal{P}$") within each of the intervals considered. For the absolute deviations, the same scale is used for the three distances. Finally, rows "2-norm" report the two-norm of the deviations (i.e., $||z||_2$), again for sensitive, nonsensitive, and all the cells.

Looking at Tables 3–9 we can draw some conclusions about the behaviour of each of the three objectives. As for performance, we see that most of the optimization problems could be solved until optimality in few seconds on a standard personal computer. For L_1 and L_∞ the best solution algorithm depends on the particular instance, and it is difficult to know in advance which will be the best choice. It is also clear that L_∞ provides the slowest executions, due to the number of extra constraints considered in (24). The L_2 objective, solved through a quadratic interior-point solver, was always the most efficient choice (except for the smallest instance hier13, where it was

Table 3: Results for the hier13 instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	3.25		6.86		3.83				5.85		35.23	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	37.8	44.1	344.0	all	33.9	33.7	313.4	all	52.0	58.2	463.7
dev.	$\in\mathcal{P}$	55.6	28.0	97.0	$\in\mathcal{P}$	55.2	27.8	97.0	$\in\mathcal{P}$	59.0	27.8	97.0
	$\notin\mathcal{P}$	36.8	44.6	344.0	$\notin\mathcal{P}$	32.7	33.6	313.4	$\notin\mathcal{P}$	51.5	59.4	463.7
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	5	673	5	0	5	277	5	0	5	597	5
	5	9	61	2	5	9	179	2	5	9	43	2
	9	23	192	11	9	23	486	11	9	23	160	11
	23	46	370	27	23	46	501	27	23	46	232	17
Distr.	46	70	257	20	46	70	267	22	46	70	249	20
abs.	70	93	142	42	70	93	82	40	70	93	271	52
dev.	93	139	149	5	93	139	85	5	93	139	186	5
	139	185	44	0	139	185	18	0	139	185	116	0
	185	232	9	0	185	232	10	0	185	232	22	0
	232	325	10	0	232	325	3	0	232	325	23	0
	325	464	1	0	325	464	0	0	325	464	9	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	0.81	1.72	9.97	all	0.87	1.95	45.84	all	1.04	1.91	9.97
dev.	$\in\mathcal{P}$	6.20	2.17	9.97	$\in\mathcal{P}$	6.18	2.19	9.97	$\in\mathcal{P}$	6.65	2.37	9.97
	$\notin\mathcal{P}$	0.49	1.02	8.28	$\notin\mathcal{P}$	0.56	1.42	45.84	$\notin\mathcal{P}$	0.71	1.25	8.28
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	847	0	0.0	0.1	537	0	0.0	0.1	720	0
	0.1	0.5	596	0	0.1	0.5	845	0	0.1	0.5	499	0
	0.5	1.0	224	0	0.5	1.0	256	0	0.5	1.0	286	0
	1.0	1.5	81	0	1.0	1.5	111	0	1.0	1.5	163	0
Distr.	1.5	2.0	55	0	1.5	2.0	67	0	1.5	2.0	79	0
perc.	2.0	5.0	79	61	2.0	5.0	68	61	2.0	5.0	122	51
dev.	5.0	10.0	26	51	5.0	10.0	23	51	5.0	10.0	39	61
	10.0	15.0	0	0	10.0	15.0	0	0	10.0	15.0	0	0
	15.0	30.0	0	0	15.0	30.0	0	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	1	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		2609.6		all		2149.3		all		3504.9	
2-norm	$\in\mathcal{P}$		658.5		$\in\mathcal{P}$		654.1		$\in\mathcal{P}$		689.3	
	$\notin\mathcal{P}$		2525.1		$\notin\mathcal{P}$		2047.4		$\notin\mathcal{P}$		3436.4	

Table 4: Results for the hier16 instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	19.85		28.36		17.19				66.52		136.86	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	35.8	40.0	280.5	all	33.4	30.6	258.3	all	36.8	36.6	300.9
dev.	$\in\mathcal{P}$	48.3	27.4	131.0	$\in\mathcal{P}$	48.3	27.4	131.0	$\in\mathcal{P}$	48.7	27.4	131.0
	$\notin\mathcal{P}$	34.9	40.6	280.5	$\notin\mathcal{P}$	32.4	30.6	258.3	$\notin\mathcal{P}$	36.0	37.0	300.9
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	3	992	7	0	3	289	7	0	3	644	7
	3	6	87	8	3	6	270	8	3	6	172	8
	6	15	333	18	6	15	620	18	6	15	440	18
	15	30	454	31	15	30	739	31	15	30	563	30
Distr.	30	45	406	34	30	45	612	34	30	45	437	35
abs.	45	60	373	53	45	60	318	53	45	60	367	51
dev.	60	90	372	58	60	90	300	58	60	90	424	60
	90	120	184	14	90	120	129	14	90	120	185	14
	120	150	71	1	120	150	40	1	120	150	66	1
	150	211	53	0	150	211	19	0	150	211	32	0
	211	301	15	0	211	301	4	0	211	301	10	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	0.83	1.84	10.00	all	0.90	1.81	10.00	all	1.13	2.05	10.00
dev.	$\in\mathcal{P}$	6.89	2.38	10.00	$\in\mathcal{P}$	6.89	2.38	10.00	$\in\mathcal{P}$	7.04	2.41	10.00
	$\notin\mathcal{P}$	0.43	0.78	7.59	$\notin\mathcal{P}$	0.50	0.75	7.59	$\notin\mathcal{P}$	0.73	1.26	7.59
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	1401	0	0.0	0.1	772	0	0.0	0.1	1059	0
	0.1	0.5	1088	0	0.1	0.5	1590	0	0.1	0.5	1063	0
	0.5	1.0	480	0	0.5	1.0	555	0	0.5	1.0	507	0
	1.0	1.5	179	0	1.0	1.5	213	0	1.0	1.5	269	0
Distr.	1.5	2.0	63	0	1.5	2.0	98	0	1.5	2.0	153	0
perc.	2.0	5.0	112	101	2.0	5.0	95	101	2.0	5.0	208	95
dev.	5.0	10.0	17	117	5.0	10.0	17	117	5.0	10.0	81	120
	10.0	15.0	0	6	10.0	15.0	0	6	10.0	15.0	0	9
	15.0	30.0	0	0	15.0	30.0	0	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		3203.5		all		2706.3		all		3098.4	
2-norm	$\in\mathcal{P}$		830.2		$\in\mathcal{P}$		830.2		$\in\mathcal{P}$		836.8	
	$\notin\mathcal{P}$		3094.1		$\notin\mathcal{P}$		2575.9		$\notin\mathcal{P}$		2983.2	

Table 5: Results for the bts4 instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	16.46		39.7		11.45				1594.69		207.02	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	33.9	89.2	4483.0	all	24.9	33.0	795.9	all	30.1	49.0	947.8
dev.	$\in\mathcal{P}$	56.0	32.2	155.0	$\in\mathcal{P}$	56.0	32.2	155.0	$\in\mathcal{P}$	57.0	32.6	168.5
	$\notin\mathcal{P}$	32.4	91.5	4483.0	$\notin\mathcal{P}$	22.9	32.0	795.9	$\notin\mathcal{P}$	28.4	49.4	947.8
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	45	26774	909	0	45	28849	910	0	45	26325	890
	45	90	3593	907	45	90	4062	907	45	90	4543	911
	90	224	3109	444	90	224	1338	443	90	224	3185	459
	224	448	637	0	224	448	58	0	224	448	229	0
Distr.	448	672	124	0	448	672	1	0	448	672	20	0
abs.	672	897	22	0	672	897	2	0	672	897	7	0
dev.	897	1345	26	0	897	1345	0	0	897	1345	1	0
	1345	1793	21	0	1345	1793	0	0	1345	1793	0	0
	1793	2242	0	0	1793	2242	0	0	1793	2242	0	0
	2242	3138	1	0	2242	3138	0	0	2242	3138	0	0
	3138	4483	2	0	3138	4483	0	0	3138	4483	0	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	0.74	1.97	11.11	all	0.84	1.95	20.23	all	1.10	2.36	11.11
dev.	$\in\mathcal{P}$	7.27	2.60	11.11	$\in\mathcal{P}$	7.27	2.59	11.11	$\in\mathcal{P}$	7.46	2.61	11.11
	$\notin\mathcal{P}$	0.31	0.83	11.03	$\notin\mathcal{P}$	0.42	0.84	20.23	$\notin\mathcal{P}$	0.68	1.63	11.03
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	23282	0	0.0	0.1	14581	0	0.0	0.1	19900	0
	0.1	0.5	5640	0	0.1	0.5	11977	0	0.1	0.5	5210	0
	0.5	1.0	2410	2	0.5	1.0	4163	2	0.5	1.0	3119	0
	1.0	1.5	1085	3	1.0	1.5	1626	3	1.0	1.5	1697	3
Distr.	1.5	2.0	652	2	1.5	2.0	738	2	1.5	2.0	1131	4
perc.	2.0	5.0	962	51	2.0	5.0	933	52	2.0	5.0	2138	46
dev.	5.0	10.0	275	1478	5.0	10.0	282	1481	5.0	10.0	774	1447
	10.0	15.0	4	724	10.0	15.0	9	720	10.0	15.0	341	760
	15.0	30.0	0	0	15.0	30.0	1	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		18243.0		all		7912.0		all		10997.2	
2-norm	$\in\mathcal{P}$		3072.3		$\in\mathcal{P}$		3070.3		$\in\mathcal{P}$		3120.5	
	$\notin\mathcal{P}$		17982.4		$\notin\mathcal{P}$		7292.0		$\notin\mathcal{P}$		10545.2	

Table 6: Results for the nine5d instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	126.67		43.03		20.36				784.52		137.33	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	41.4	68.8	1010.0	all	37.2	37.5	499.4	all	34.4	38.4	306.5
dev.	$\in\mathcal{P}$	50.6	29.3	156.0	$\in\mathcal{P}$	50.6	29.3	156.0	$\in\mathcal{P}$	50.8	29.3	156.0
	$\notin\mathcal{P}$	39.7	73.6	1010.0	$\notin\mathcal{P}$	34.7	38.3	499.4	$\notin\mathcal{P}$	31.4	39.1	306.5
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	10	4884	177	0	10	2082	177	0	10	3901	174
	10	20	395	187	10	20	1841	187	10	20	943	185
	20	51	1268	464	20	51	3285	464	20	51	2027	469
	51	101	1518	822	51	101	1337	822	51	101	1595	814
Distr.	101	152	498	10	101	152	351	10	101	152	478	18
abs.	152	202	236	1	152	202	122	1	152	202	107	1
dev.	202	303	156	0	202	303	38	0	202	303	20	0
	303	404	61	0	303	404	12	0	303	404	1	0
	404	505	13	0	404	505	4	0	404	505	0	0
	505	707	32	0	505	707	0	0	505	707	0	0
	707	1010	10	0	707	1010	0	0	707	1010	0	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	1.67	2.69	10.00	all	1.90	2.53	10.00	all	2.23	3.02	10.00
dev.	$\in\mathcal{P}$	6.83	2.42	10.00	$\in\mathcal{P}$	6.83	2.42	10.00	$\in\mathcal{P}$	6.87	2.42	10.00
	$\notin\mathcal{P}$	0.73	1.31	9.78	$\notin\mathcal{P}$	1.00	1.11	9.31	$\notin\mathcal{P}$	1.38	2.25	8.79
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	4820	0	0.0	0.1	831	0	0.0	0.1	3400	0
	0.1	0.5	1208	0	0.1	0.5	2572	0	0.1	0.5	1503	0
	0.5	1.0	855	1	0.5	1.0	2668	0	0.5	1.0	1050	0
	1.0	1.5	675	2	1.0	1.5	1386	3	1.0	1.5	775	2
Distr.	1.5	2.0	373	1	1.5	2.0	569	1	1.5	2.0	399	1
perc.	2.0	5.0	978	923	2.0	5.0	918	923	2.0	5.0	1192	889
dev.	5.0	10.0	163	664	5.0	10.0	128	664	5.0	10.0	753	682
	10.0	15.0	0	70	10.0	15.0	0	70	10.0	15.0	0	87
	15.0	30.0	0	0	15.0	30.0	0	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		8316.4		all		5468.3		all		5343.4	
2-norm	$\in\mathcal{P}$		2383.6		$\in\mathcal{P}$		2383.2		$\in\mathcal{P}$		2389.6	
	$\notin\mathcal{P}$		7967.5		$\notin\mathcal{P}$		4921.7		$\notin\mathcal{P}$		4779.4	

Table 7: Results for the ninenew instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	27.08		24.02		11.15				199.39		120.52	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	41.6	53.0	602.7	all	38.6	39.0	522.8	all	39.0	43.2	439.1
dev.	$\in\mathcal{P}$	52.4	28.6	192.0	$\in\mathcal{P}$	52.4	28.3	192.0	$\in\mathcal{P}$	53.0	28.3	192.0
	$\notin\mathcal{P}$	39.9	55.5	602.7	$\notin\mathcal{P}$	36.6	40.0	522.8	$\notin\mathcal{P}$	36.8	44.7	439.1
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	6	2287	23	0	6	896	23	0	6	1858	19
	6	12	266	66	6	12	710	64	6	12	417	65
	12	30	698	120	12	30	1641	119	12	30	967	116
	30	60	932	287	30	60	1434	291	30	60	1135	289
Distr.	60	90	682	269	60	90	540	271	60	90	630	277
abs.	90	121	367	88	90	121	211	87	90	121	375	89
dev.	121	181	296	4	121	181	190	2	121	181	234	2
	181	241	96	1	181	241	48	1	181	241	55	1
	241	301	37	0	241	301	9	0	241	301	8	0
	301	422	20	0	301	422	6	0	301	422	8	0
	422	603	7	0	422	603	3	0	422	603	1	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	1.56	2.47	16.16	all	1.76	2.44	22.86	all	2.19	2.93	10.00
dev.	$\in\mathcal{P}$	6.66	2.38	10.00	$\in\mathcal{P}$	6.66	2.36	10.00	$\in\mathcal{P}$	6.79	2.39	10.00
	$\notin\mathcal{P}$	0.79	1.29	16.16	$\notin\mathcal{P}$	1.02	1.35	22.86	$\notin\mathcal{P}$	1.50	2.32	9.93
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	2368	0	0.0	0.1	493	0	0.0	0.1	1763	0
	0.1	0.5	975	0	0.1	0.5	1829	0	0.1	0.5	1013	0
	0.5	1.0	888	2	0.5	1.0	1606	0	0.5	1.0	820	0
	1.0	1.5	501	0	1.0	1.5	733	0	1.0	1.5	503	0
Distr.	1.5	2.0	318	0	1.5	2.0	386	0	1.5	2.0	309	0
perc.	2.0	5.0	531	509	2.0	5.0	510	507	2.0	5.0	793	479
dev.	5.0	10.0	104	315	5.0	10.0	119	319	5.0	10.0	487	329
	10.0	15.0	2	32	10.0	15.0	10	32	10.0	15.0	0	50
	15.0	30.0	1	0	15.0	30.0	2	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		5447.5		all		4444.3		all		4708.1	
2-norm	$\in\mathcal{P}$		1749.6		$\in\mathcal{P}$		1744.5		$\in\mathcal{P}$		1759.5	
	$\notin\mathcal{P}$		5158.9		$\notin\mathcal{P}$		4087.6		$\notin\mathcal{P}$		4366.9	

Table 8: Results for the two5in6 instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	13.58		16.88		9				83.48		86.47	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	38.3	52.8	530.0	all	35.4	34.9	340.1	all	38.3	39.3	281.8
dev.	$\in\mathcal{P}$	49.1	32.0	169.0	$\in\mathcal{P}$	49.1	32.0	169.0	$\in\mathcal{P}$	49.7	31.8	169.0
	$\notin\mathcal{P}$	36.7	55.0	530.0	$\notin\mathcal{P}$	33.5	34.9	340.1	$\notin\mathcal{P}$	36.7	40.0	281.8
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	5	2364	46	0	5	621	46	0	5	1487	46
	5	11	221	55	5	11	723	55	5	11	334	42
	11	27	470	140	11	27	1453	140	11	27	698	143
	27	53	516	155	27	53	1190	155	27	53	1099	163
Distr.	53	80	510	161	53	80	528	161	53	80	604	161
abs.	80	106	409	159	80	106	210	159	80	106	411	161
dev.	106	159	267	0	106	159	176	0	106	159	266	0
	159	212	111	4	159	212	45	4	159	212	53	4
	212	265	66	0	212	265	11	0	212	265	5	0
	265	371	23	0	265	371	4	0	265	371	4	0
	371	530	3	0	371	530	0	0	371	530	0	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	1.46	2.49	10.00	all	1.65	2.40	17.88	all	2.08	2.81	10.00
dev.	$\in\mathcal{P}$	6.80	2.42	10.00	$\in\mathcal{P}$	6.80	2.42	10.00	$\in\mathcal{P}$	6.99	2.42	10.00
	$\notin\mathcal{P}$	0.69	1.23	9.69	$\notin\mathcal{P}$	0.90	1.17	17.88	$\notin\mathcal{P}$	1.37	2.04	8.44
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	2495	0	0.0	0.1	394	0	0.0	0.1	1563	0
	0.1	0.5	765	0	0.1	0.5	1780	0	0.1	0.5	923	0
	0.5	1.0	567	0	0.5	1.0	1525	0	0.5	1.0	606	0
	1.0	1.5	360	3	1.0	1.5	533	3	1.0	1.5	482	3
Distr.	1.5	2.0	260	0	1.5	2.0	259	0	1.5	2.0	310	0
perc.	2.0	5.0	424	374	2.0	5.0	384	374	2.0	5.0	713	339
dev.	5.0	10.0	90	322	5.0	10.0	83	322	5.0	10.0	364	338
	10.0	15.0	0	21	10.0	15.0	0	21	10.0	15.0	0	40
	15.0	30.0	0	0	15.0	30.0	3	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		4917.2		all		3749.3		all		4137.1	
2-norm	$\in\mathcal{P}$		1573.0		$\in\mathcal{P}$		1572.0		$\in\mathcal{P}$		1582.4	
	$\notin\mathcal{P}$		4658.8		$\notin\mathcal{P}$		3403.8		$\notin\mathcal{P}$		3822.5	

Table 9: Results for the nine12 instance

	L_1				L_2				L_∞			
	Simplex		Int. Point		Int. Point				Simplex		Int. Point	
CPU	382.13		47.38		18.29				727.28		338.8	
		mean	std	max.		mean	std	max.		mean	std	max.
Abs.	all	36.3	44.3	490.9	all	34.6	33.0	377.4	all	32.6	36.5	268.0
dev.	$\in\mathcal{P}$	51.7	28.3	154.0	$\in\mathcal{P}$	51.6	28.2	154.0	$\in\mathcal{P}$	52.1	28.2	154.0
	$\notin\mathcal{P}$	34.4	45.6	490.9	$\notin\mathcal{P}$	32.4	33.0	377.4	$\notin\mathcal{P}$	30.1	36.7	268.0
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0	5	3662	17	0	5	1280	17	0	5	3184	12
	5	10	450	82	5	10	1101	82	5	10	668	83
	10	25	1050	155	10	25	2589	155	10	25	1546	149
	25	49	1561	313	25	49	2282	313	25	49	1644	321
Distr.	49	74	979	310	49	74	1042	314	49	74	1033	307
abs.	74	98	659	273	74	98	473	269	74	98	567	276
dev.	98	147	566	26	98	147	336	26	98	147	451	28
	147	196	203	2	147	196	96	2	147	196	110	2
	196	245	64	0	196	245	15	0	196	245	16	0
	245	344	25	0	245	344	5	0	245	344	2	0
	344	491	2	0	344	491	2	0	344	491	0	0
		mean	std	max.		mean	std	max.		mean	std	max.
Perc.	all	1.35	2.34	12.55	all	1.53	2.32	25.43	all	1.74	2.64	10.00
dev.	$\in\mathcal{P}$	6.71	2.38	10.00	$\in\mathcal{P}$	6.70	2.39	11.97	$\in\mathcal{P}$	6.82	2.40	10.00
	$\notin\mathcal{P}$	0.67	1.15	12.55	$\notin\mathcal{P}$	0.87	1.23	25.43	$\notin\mathcal{P}$	1.09	1.85	8.95
	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$	from	to	$\notin\mathcal{P}$	$\in\mathcal{P}$
	0.0	0.1	3892	0	0.0	0.1	1036	0	0.0	0.1	3285	0
	0.1	0.5	1967	0	0.1	0.5	3489	0	0.1	0.5	1876	0
	0.5	1.0	1376	0	0.5	1.0	2341	0	0.5	1.0	1371	0
	1.0	1.5	770	0	1.0	1.5	1044	0	1.0	1.5	763	0
Distr.	1.5	2.0	401	3	1.5	2.0	453	3	1.5	2.0	472	3
perc.	2.0	5.0	695	676	2.0	5.0	709	678	2.0	5.0	971	636
dev.	5.0	10.0	116	452	5.0	10.0	131	449	5.0	10.0	483	475
	10.0	15.0	4	47	10.0	15.0	15	48	10.0	15.0	0	64
	15.0	30.0	0	0	15.0	30.0	3	0	15.0	30.0	0	0
	30.0	50.0	0	0	30.0	50.0	0	0	30.0	50.0	0	0
	50.0	100.0	0	0	50.0	100.0	0	0	50.0	100.0	0	0
	all		5840.1		all		4878.1		all		4988.1	
2-norm	$\in\mathcal{P}$		2022.2		$\in\mathcal{P}$		2017.2		$\in\mathcal{P}$		2034.2	
	$\notin\mathcal{P}$		5478.8		$\notin\mathcal{P}$		4441.5		$\notin\mathcal{P}$		4554.5	

only 0.6 seconds slower than the L_1 and simplex combination). In most instances the solution time of the L_2 objective was about half the time of the second fastest option. This is because, first, the complexity of solving a quadratic separable optimization problem (i.e., with a diagonal weight matrix $\mathbf{W}$) is the same that for a linear one, if we use an interior-point algorithm; and second, problem (22) involves the double of variables that (23). It is also worth to note that the solution times obtained with the interior-point algorithm, for the three objectives, can even be improved using specialized solvers that exploit the tables structure. Some work has already been done along these lines for very large three-dimensional tables (Castro 2003) using specialized interior-point algorithms (Castro 2000).

The L_2 objective provides also the lowest means and, mainly, the lowest standard deviations for the absolute deviations. Such lowest standard deviations are not surprising, since the L_2 objective, due to its quadratic nature, attempts to evenly distribute the required deviations among all the cells. As for the other two objectives, L_∞ provided better absolute deviations than L_1, but for instances hier13 and hier16. That was, a priori, an unexpected result, since only two cells appear in the objective function of (24), whereas all the perturbations are considered in (22). The distribution of the absolute deviations shows that L_1 provides the greater number of cells in the lowest interval. However, L_2 reports less cells with medium-large deviations than the other two distances. This is because such large deviations are highly (i.e., quadratically) penalized in the L_2 objective function.

As for the percentage deviations, L_1 must clearly provide the best mean values, since its objective function is exactly the sum of percentage absolute deviations (as said before, we used weights $w_i = 1/a_i$). However, the L_2 objective provides similar mean percentage deviations, and, for most instances, with slightly better standard deviations. L_∞ provided worser means and standard deviations, but, as a consequence of its objective function, the lowest maximum values. The best mean values of the L_1 objective are observed in the distribution of the percentage deviations: most values happen to be in the lowest intervals. Although that is also true for the other two objectives, they show a different distribution pattern. L_2 tends to distribute the values for all the intervals (thus reducing the number of points in the first one [0.0%, 0.1%]), whereas L_∞ permits a significant number of values with medium percentage deviations (since it only focuses in the largest one). On the other hand, if we look at the largest intervals, L_2 provides in most cases the lowest number of points greater than, e.g., 2.0%.

Finally, the lowest two-norms of the deviations vector are provided in all the instances by the L_2 objective. This is a consequence of L_2 being the only quadratic objective of the three tested. Except for instance hier13, L_∞ always provides deviations with better two-norms than L_1.

From the above comments, we can conclude that the L_1 objective provides the best results when a first-order comparison measure, as the mean percentage deviation, is considered. However, when a second-order measure is used, as the two-norm of the deviations or the standard deviation of the percentage deviations, L_2 seems to be the best choice. The above is an immediate result of the objective functions (linear or quadratic) of the respective optimization problems. As for the distribution of absolute and percentage deviations, L_1 provides more cells in the lowest interval, but also with more medium-large deviations than L_2; the latter distributes more uniformly the deviations among all the cells. Computationally, the fastest option is the L_2 objective. L_∞ provides acceptable results for both the first-order and second-order comparison measures. However, it is computationally expensive, which makes it a less convenient choice for large volumes of data. The distances can be combined into a single objective function to meet the end-user requirements (e.g., L_1 for internal and L_2—possibly with a penalty parameter—for marginal cells).

8.2 Solving the CSPLIB instances

For this group of experiments we omitted the seven complex instances of last Subsection, and those involving a small number of cells. Table 10 shows the features of the instances considered. Columns "Name", "n", "$|\mathcal{P}|$" and "m" have the same meaning that in Table 2. Column "N.coef" gives the number of coefficients of the constraints matrix $\mathbf{M}$. Table 11 shows the results obtained with L_1, L_2 and L_∞. For each distance, the execution time (columns "CPU"), average percentage deviation for all the cells (columns "$\overline{\%\text{Dev.}}$"), and two-norm of the deviations vector (columns "2-norm") are provided. The results reported for L_∞ were computed by the simplex method: as stated in Subsection 8.1, the interior-point solutions, although with the same objective function, provided worst average percentage deviations and distances for all the instances. The results for L_1 with the simplex and interior-point method were similar, although the simplex was the most efficient choice in most cases. Those are the results reported in the Table, but for the four instances which are clearly marked. In three of these four cases, the simplex method provided a wrong solution. Tuning CPLEX 8.0 we were able to solve them. The interior-point method could solve all the instances with the default settings.

Most of the conclusions drawn in Subsection 8.1 also apply here: L_1 and L_2 provide the best results for, respectively, first and second order measures, and L_∞ the slowest executions. The end-user can choose the most appropriate distance for its particular data. Suitable choices are L_1 if a number of cells with small percentage deviations is required, or L_2 if the goal is to reduce the two-norm between the original and perturbed values. Figure 4 shows the effect of both distances on a very small one-dimensional table. The table considered is $a_1 + a_2 = a_3$, with $a_1 = 12$ and $a_2 = 8$. We imposed $z_1 + z_2 = z_3$ and $z_3 \geq 4$, i.e., an upper protection level of 4 is forced for the marginal sensitive cell. Using $w_i = 1/a_i$ the optimal solution obtained with L_1 is $z_1 = 4$, $z_2 = 0$ and $z_3 = 4$. With the same weights, the optimal solution provided by L_2 is $z_1 = 2.4$, $z_2 = 1.6$ and $z_3 = 4$. If integer values were required, the z_1 and z_2 values could be rounded through some heuristic postprocess (in that case the most reasonable choice would be $z_1 = 2$ and $z_2 = 2$). Both distances can be combined in the single objective $\omega(\sum_{i=1}^n w_i|z_i|) + (1-\omega)(\sum_{i=1}^n w_i z_i^2)$, $\omega \in [0,1]$ being a weight for the linear and quadratic terms. For $\omega = 1$ and $\omega = 0$ the combined objective corresponds to the L_1 and L_2 distances, respectively. Figure 4 shows the perturbed internal cell values obtained for $\omega = 0, 0.1, \ldots, 0.9, 1$, and the original ones (a_1, a_2). Clearly, the L_2 point is closer to $(12, 8)$, but the L_1 solution preserves the value of cell a_2. This is consistent with the results observed for the CSPLIB instances.

9 Conclusions

The minimum-distance controlled perturbation framework introduced in that work proved to be a promising tool for tabular data protection. We examined three particular methods, using the L_1, L_2 and L_∞ distances. The L_1 variant was independently suggested, using an alternative derivation, by Dandekar and Cox (2002). The minimum-distance approach has shown to be efficient: can solve real-world large problems in few seconds; versatile: deals with any table or set of tables, and with any additional linear constraint (e.g., preserving the value of marginal cells); and safe: even with partial information, an attacker is not able to reproduce the original data. Alternative approaches for tabular data protection have flaws in some of the above features.

The three methods tested, for L_1, L_2 and L_∞, provided different patterns of deviations, each of

Table 10: Dimensions of the largest CSPLIB instances

Name	n	$\|\mathcal{P}\|$	m	N.coef
cbs	11163	2467	244	22326
dale	16514	4923	405	33028
hier13x13x13a	2197	108	3549	11661
hier13x13x13b	2197	108	3549	11661
hier13x13x13c	2197	108	3549	11661
hier13x13x13d	2197	108	3549	11661
hier13x13x13e	2197	112	3549	11661
hier13x13x7d	1183	75	1443	5369
hier13x7x7d	637	50	525	2401
hier16x16x16a	4096	224	5376	21504
hier16x16x16b	4096	224	5376	21504
hier16x16x16c	4096	224	5376	21504
hier16x16x16d	4096	224	5376	21504
hier16x16x16e	4096	224	5376	21504
jjtabeltest3	3025	1054	1650	7590
osorio	10201	7	202	20402
table1	1584	146	510	4752
table3	4992	517	2464	19968
table4	4992	517	2464	19968
table5	4992	517	2464	19968
table6	1584	146	510	4752
table7	624	17	230	1872
table8	1271	3	72	2542
targus	162	13	63	360
toy3dsarah	2890	376	1649	9690

Table 11: Results for the largest CSPLIB instances

	L_1			L_2			L_∞		
name	CPU	%Dev.	2-norm	CPU	%Dev.	2-norm	CPU	%Dev.	2-norm
cbs	0.0	40.6	75986	0.1	42.9	55732	0.1	40.6	75986
dale	0.7	18.7	4991	0.3	20.3	1859	1.5	21.1	3086
hier13x13x13a	1.9	0.8	3094	2.4	0.9	2162	5.9	1.0	3201
hier13x13x13b	2.0	0.8	3094	2.3	0.9	2162	5.7	1.0	3201
hier13x13x13c	1.9	0.8	3094	2.5	0.9	2162	5.7	1.0	3201
hier13x13x13d	2.5	1.6	6187	2.4	1.7	4323	2.5	2.1	7182
hier13x13x13e	2.5	1.6	6187	2.4	1.7	4323	2.6	2.1	6493
hier13x13x7d	0.2	0.8	2431	0.3	0.9	1463	0.5	1.1	2588
hier13x7x7d	0.0	0.9	1850	0.1	1.0	1075	0.1	1.1	2143
hier16x16x16a	4.6	0.8	4868	12.0	0.9	2796	33.1	1.0	6053
hier16x16x16b	4.7	0.8	4868	12.1	0.9	2796	32.9	1.0	6053
hier16x16x16c	4.7	0.8	4868	12.0	0.9	2796	33.1	1.0	6053
hier16x16x16d	5.3	1.6	9737	12.0	1.8	5593	46.7	2.2	9337
hier16x16x16e	5.3	1.6	9737	12.0	1.8	5593	46.9	2.2	9337
jjtabeltest3	0.2	22.1	3.4e+7	0.1	27.8	2.0e+7	0.2	30.0	2.7e+7
osorio	0.1	0.0	0	0.2	0.0	0	15.8	0.0	0
table1	[1]0.2	0.9	5.2e+6	0.0	1.1	2.5e+6	0.1	1.1	5.3e+6
table3	0.9	3.0	162763	0.7	3.5	72291	12.7	3.8	111104
table4	0.9	3.0	162763	0.7	3.5	72291	12.6	3.8	111104
table5	1.0	3.0	162763	0.7	3.5	72291	12.6	3.8	111104
table6	[1]0.3	0.9	4.1e+6	0.0	1.1	2.5e+7	0.1	1.1	5.3e+6
table7	0.0	5.9	50738	0.0	7.2	32984	0.0	7.5	50122
table8	0.0	0.0	26	0.0	0.1	15	0.1	0.1	19
targus	[2]0.0	4.1	6958	0.0	4.1	4964	0.0	4.1	6961
toy3dsarah	[1]0.1	2.7	2.4e+10	0.1	3.0	2.3e+10	0.0	2.8	2.4e+10

[1] Simplex provided a wrong solution; interior-point one used

[2] Best results obtained with the interior-point algorithm

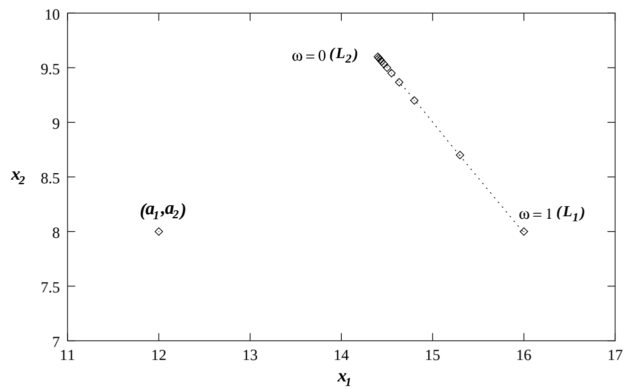

Figure 4: Solutions of the L_1 and L_2 distances for the one dimensional table $a_1+a_2 = a_3$, imposing a perturbation $z_3 \geq 4$ for the marginal cell. Point $(a_1, a_2) = (12, 8)$ corresponds to the original internal cell values. The other eleven points are the solutions obtained with the objective function $\omega(\sum_{i=1}^{n} w_i|z_i|) + (1-\omega)(\sum_{i=1}^{n} w_i z_i^2)$, for $\omega = 0, 0.1, 0.2, \ldots, 0.9, 1$, which combines the L_1 and L_2 distances through the weight factor ω. The L_2 solution (computed with $\omega = 0$) is closer to (a_1, a_2), but the L_1 point ($\omega = 1$) preserves the value of a_2.

them with a clear behaviour. National Statistical Agencies would choose the best suited method for their data. It is also possible to combine them, mainly L_1 and L_2, to fit particular needs.

Some related fields of research can be explored. One of them is to deal with frequency tables. Except for particular situations, as, e.g., two-dimensional tables and the L_1 distance, the deviations computed can have fractional values, and thus not being valid for an integer table. There are two ways to obtain integer deviations. The most efficient one is to produce them from the fractional solution computed by the methods presented in this work. A heuristic post-process should be used for this purpose. The second possibility is to solve an integer programming problem (e.g., forcing integer deviations in the optimization problems of this work). In general, for large tables, that can result in impractical execution times.

A second field of research deals with the optimization solvers. In a static environment, the final goal might be the protection, in a single run, of all the tables derived from the same microdata. The resulting problem is huge. In a dynamic environment, the goal would be the online protection of particular tables (e.g., obtained from end-user queries from a data-warehouse). Speed is instrumental in that case. In both situations, we may need highly-efficient implementations of the optimization methods used in this work, which exploit the problem structure. Some work has already been done in this direction for large (i.e., one million cells) three-dimensional tables and L_2 (Castro 2000, 2003), where a specialized implementation was two orders of magnitude faster than the CPLEX 8.0 solver.

References

Bacharach, M. (1966), "Matrix Rounding Problems," *Management Science*, 9, 732–742.

Bixby R. E. (2002), "Solving Real-World Linear Programs: a Decade and More of Progress," *Operations Research*, 50, 3–15.

Carvalho, F.D., Dellaert, N.P., and Osório, M.D. (1994), "Statistical Disclosure in Two-Dimensional Tables: General Tables," *Journal of the American Statistical Association*, 89, 1547–1557.

Castro, J. (2000), "A Specialized Interior-Point Algorithm for Multicommodity Network Flows," *SIAM Journal on Optimization*, 10(3), 852–877.

Castro, J. (2002), "Network Flows Heuristics for Complementary Cell Suppression: an Empirical Evaluation and Extensions," in *Lecture Notes in Computer Science. Inference Control in Statistical Databases* (Vol. 2316), ed. J. Domingo-Ferrer, Berlin: Springer, 59–73.

Castro, J. (2003), "Quadratic Interior-Point Methods in Statistical Disclosure Control," Research Report 2003-10, Universitat Politècnica de Catalunya, Dept. of Statistics and Operations Research. Available from `http://www-eio.upc.es/~jcastro`.

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

Cox, L.H. (1995), "Network Models for Complementary Cell Suppression," *Journal of the American Statistical Association*, 90, 1453–1462.

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

Cox, L. H., and George, J. A. (1989), "Controlled Rounding for Tables with Subtotals," *Annals of Operations Research*, 20, 141–157.

Dandekar, R.A. (2003), "Cost Effective implementation of Synthetic Tabulation (a.k.a. Controlled Tabular Adjustments) in Legacy and New Statistical Data Publication Systems," presented at the Joint ECE/Eurostat Work Session on Statistical Data Confidentiality, Luxembourg. Available online from `http://www.unece.org/stats/documents/2003.04.confidentiality.htm`.

Dandekar, R.A., and Cox, L.H. (2002), "Synthetic Tabular Data: an Alternative to Complementary Cell Suppression," unpublished manuscript, Energy Information Administration, U.S. Department of Energy. Available from the first author on request (`Ramesh.Dandekar@eia.doe.gov`).

Dantzig, G.B. (1963), *Linear Programming and Extensions*, Princeton: Princeton University Press.

Dellaert, N.P., and Luijten, W.A. (1999), "Statistical Disclosure in General Three-Dimensional Tables," *Statistica Neerlandica*, 53, 197–221.

Domingo-Ferrer, J. (ed.) (2002), *Lecture Notes in Computer Science. Inference Control in Statistical Databases* (Vol. 2316), Berlin: Springer.

Domingo-Ferrer, J., and Torra V. (2002), "A Critique of the Sensitivity Rules Usually Employed for Statistical Table Protection," *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, 10(5), 545–556.

Fischetti, M., and Salazar, J.J. (2000), "Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints," *Journal of the American Statistical Association*, 95, 916–928.

Fourer, R, Gay, D.M., and Kernighan, B.W. (1993), *AMPL: A Modeling Language for Mathematical Programming*, Danvers, MA: Boyd & Fraser.

ILOG CPLEX (2002), *ILOG CPLEX 8.0 Reference Manual Library*, Gentilly, France: ILOG.

Kelly, J. P., Assad, A. A., and Golden, B. L. (1990), "The Controlled Rounding Problem: Relaxations and Complexity Issues," *OR Spektrum*, 12, 129–138.

Kelly, J. P., Golden, B. L., and Assad, A. A. (1990), "Using Simulated Annealing to Solve Controlled Rounding Problems," *Annals of Operations Research*, 2(2), 174–190.

Kelly, J.P., Golden, B.L, and Assad, A.A. (1992), "Cell Suppression: Disclosure Protection for Sensitive Tabular Data," *Networks*, 22, 28–55.

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

Luenberger, D.G. (1989), *Linear and Nonlinear Programming* (2nd ed.), Reading, MA: Addison Wesley.

Robertson, D.A., and Ethier, R. , "Cell Suppression: Experience and Theory," in *Lecture Notes in Computer Science. Inference Control in Statistical Databases* (Vol. 2316), ed. J. Domingo-Ferrer, Berlin: Springer, 8–20.

Willenborg, L., and de Waal, T. (eds.) (2000) *Lecture Notes in Statistics. Elements of Statistical Disclosure Control* (Vol. 155), New York: Springer.

Wright, S.J. (1997), *Primal-Dual Interior-Point Methods*, Philadelphia: SIAM.