Statistical disclosure control in tabular data

Jordi Castro
Dept. of Statistics and Operations Research
Universitat Politècnica de Catalunya
Jordi Girona 1–3, 08034 Barcelona, Catalonia
jordi.castro@upc.edu

Report DR 2009-11
November 2009

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

Statistical disclosure control in tabular data [1]

Jordi Castro

Abstract Data disseminated by National Statistical Agencies (NSAs) can be classified as either microdata or tabular data. Tabular data is obtained from microdata by crossing one or more categorical variables. Although cell tables provide aggregated information, they also need to be protected. This chapter is a short introduction to tabular data protection. It contains three main sections. The first one shows the different types of tables that can be obtained, and how they are modeled. The second describes the practical rules for detection of sensitive cells that are used by NSAs. Finally, an overview of protection methods is provided, with a particular focus on two of them: "cell suppression problem" and "controlled tabular adjustment".

1 Introduction

National Statistical Agencies (NSAs) store information about individuals or *respondents* (persons, companies, etc.) in microdata files. A microdata file V of s individuals and t variables is a $s \times t$ matrix where V_{ij} is the value of variable j for individual i. Formally, it can be defined as a function

$$V : I \rightarrow D(V_1) \times D(V_2) \times \cdots \times D(V_t)$$

that maps individuals of set I to an array of t values for variables $V_1,\ldots,V_t$, $D()$ being the domain of those variables. According to this domain, variables can be classified as numerical (e.g., "age", "net profit") or categorical ("sex","economy sector").

From those microdata files, tabular data is obtained by crossing one or more categorical variables. For instance, assuming a microdata file with information of

Jordi Castro
Department of Statistics and Operations Research, Universitat Politècnica de Catalunya, c. Jordi Girona 1–3, 08034 Barcelona, Catalonia, e-mail: jordi.castro@upc.edu

[1] To appear as a chapter of *Privacy and Anonymity in Information Management Systems*, Springer

inhabitants of some region, and considering only the categorical variable "profession", we could get the one-dimensional table of Figure 1. Crossing variables "profession" and "municipality" we could get the two-dimensional table of Figure 2. The above two tables count the number of inhabitants in each cell; these are named *frequency* tables. Instead, the table could provide information about a third variable. For instance, the table of Figure 3 shows the overall salary for each profession and municipality; these are named *magnitude* tables. Formally, a table is a function

$$T : D(V_{i_1}) \times D(V_{i_2}) \times \cdots \times D(V_{i_l}) \rightarrow \mathbb{R} \text{ or } \mathbb{N},$$

l being the number of categorical variables that were crossed. The result of function T (cells values) belongs to $\mathbb{N}$ for a frequency table, and to $\mathbb{R}$ for a magnitude table.

Fig. 1 One-dimensional frequency table showing number of persons for each profession.

P_1	P_2	P_3	P_4	P_5	TOTAL
130	73	46	90	31	370

Fig. 2 Two-dimensional frequency table showing number of persons for each profession and municipality.

	P_1	P_2	P_3	P_4	P_5	TOTAL
M_1	20	15	30	20	10	95
M_2	72	20	1	30	10	133
M_3	38	38	15	40	11	142
TOTAL	130	73	46	90	31	370

Fig. 3 Two-dimensional magnitude table showing overall salary (in 1000€) for each profession and municipality.

	P_1	P_2	P_3	P_4	P_5	TOTAL
M_1	360	450	720	400	360	2290
M_2	1440	540	22	570	320	2892
M_3	722	1178	375	800	363	3438
TOTAL	2522	2168	1117	1770	1043	8620

Although tabular data show aggregated information, there is a risk of disclosing individual information. This can be easily seen from the tables of Figures 2 and 3. Any attacker knows that the salary of the unique respondent of cell (M_2, P_3) is 22000€. This is named an *external attacker*. It there were two respondents in that cell, then any of them could deduce the other's salary, becoming an *internal attacker*. Even if there was a larger number of respondents, e.g. 5, if one of them had a salary of, e.g. 18, there would be a disclosure risk. This scenario is named *internal attack with dominance*.

The number of registers in a microdata file r is in general much larger than the number of cells n in a table ($r \gg n \gg 0$). It could be thought that, therefore, the

protection of microdata is more complex, since it involves a larger number of information. However, tabular data involve a number of linear constraints m; this linear constraints model the relations between inner and total cells, the most usual relation being that the sum of some inner cells is equal to some marginal cell. Microdata protection in general involves few (if not 0) linear constrains, and usually $m \gg 0$. For this reason, tabular data protection methods need Linear Programming (LP) and Mixed Integer Linear Programming (MILP) technology, making the protection of complex and large tables a difficult problem.

Although it contains some references to recent literature, this chapter can not be considered a comprehensive survey on statistical disclosure control of tabular data. Additional information can be found, for instance, in the research monographs [20, 21, 22, 33] and the recent survey [32]. Details about practical aspects of tabular data protection can be found in the handbook [28].

This chapter is made of three main sections, associated to the three stages of the tabular data protection process. Section 2 shows the different types of tables that can be obtained, and how they are modeled. Section 3 introduces some sensitivity rules for detection of sensitivity cells to be protected. Finally, Section 4 introduces some of the most widely used tabular data protection methods, mainly focusing on two of them, the *cell suppression problem* and the *controlled tabular adjustment*.

2 Tabular data: types and modelling

The first stage of the tabular data protection process is to know the type of table to be protected, and how to model it. It is an important stage, since some protection methods of Section 4 can be specialized (i.e., made more efficient) for some particular classes of tables.

2.1 Classification of tables

Broadly, tables can be classified, according to different criteria as follows:

2.1.1 According to the cell values

The two types of tables were already introduced in Section 1. They are:

- **Frequency tables**, also named contingency tables. They count the number of respondents that belong to each cell. Cell values are in $\mathbb{N}$
- **Magnitude tables**. They provide information about each cell respondents for another variable of the microfile. Cell values are in $\mathbb{R}$.

2.1.2 According to the sign of cell values

Protection methods usually involve the solution of difficult LP or MILP problems. The lower bounds of the variables in those problems (either 0 or $-\infty$) are usually related to the sign of the cell values. We have two cases:

- **Positive tables:** Cell values are ≥ 0. It is the most usual situation. For instance, all frequency tables and most of magnitude tables, like "salary" for "profession"$\times$ "municipality", are positive tables.
- **General tables:** Cell values can be either positive or negative. An example of general table would be "variation of gross domestic product" for "year" $\times$ "state".

2.1.3 According to table structure

This is likely the most important classification. Some protection methods can only be applied to some of the below classes of tables.

- **Single k-dimensional table:** Single table obtained by crossing k categorical variables. All the tables shown above are k-dimensional tables ($k = 1$ for the table of Figure 1, $k = 2$ for the tables of Figures 2–3). Note that the number of cells grows very quickly (exponentially) with k.
- **Hierarchical tables:** Set of tables obtained by crossing some variables, and a number of these variables have a hierarchical relation. For instance, consider the three tables of Figure 4. The left subtable shows number of respondents for "region"$\times$ "profession"; the middle subtable, a "zoom in" of region R_2, provides the number of respondents for "municipality" (of region R_2) $\times$ "profession"; finally the right subtable, "zip code" $\times$ "profession", details municipality R_{21}. This table belongs to a particular class named 1H2D, two-dimensional tables with one hierarchical variable.

Fig. 4 Hierarchical table made of three subtables: "region" $\times$ "profession", "municipality" $\times$ "profession" and "zip code" $\times$ "profession"

	C_1	C_2	C_3
R_1	5	6	11
R_2	10	15	25
R_3	15	21	36

T_1

	C_1	C_2	C_3
R_{21}	8	10	18
R_{22}	2	5	7
R_2	10	15	25

T_2

	C_1	C_2	C_3
R_{211}	6	6	12
R_{212}	2	4	6
R_{21}	8	10	18

T_3

- **Linked tables:** It is the most general situation. Linked tables is a set of tables obtained from the same microdata file. In theory, the set of all tables obtained from a microdata file should be considered together as a (likely huge) linked table. Hierarchical and k-dimensional tables are particular cases of linked tables.

Note that, in theory, the only safe way for protecting all the tables from a microfile, is to jointly protect them as a single linked table. Unfortunately, in many cases the size of the resulting table would be excessive for current LP or MILP technology.

2.2 Modelling tables

Since linked tables are the more general case, a model for them is valid for all types of tables. However we will exploit the particular structure of two-dimensional, three-dimensional and 1H2D tables.

2.2.1 Two-dimensional tables

A two-dimensional table of $r+1$ rows and $c+1$ columns as that of Figure 5 is modeled by the following constraints.

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

Constraints (1) can be represented by the bipartite network of Figure 6. This allows the application of efficient network optimization algorithms, such as those for minimum-cost network flows, or shortest-paths [1]. This fact was originally noticed in [2], and it has been extensively used in other works [4, 5, 6, 9, 14, 23, 29].

Fig. 5 General two-dimensional table.

a_{11} ... a_{1c}		$a_{1(c+1)}$
... ...		...
a_{r1} ... a_{rc}		$a_{r(c+1)}$
$a_{(r+1)1}$ ... $a_{(r+1)c}$		$a_{(r+1)(c+1)}$

2.2.2 Three-dimensional tables

The linear constraints of a three-dimensional table of $r+1$ rows, $c+1$ columns and $l+1$ levels (levels refer to categories of third variable) are

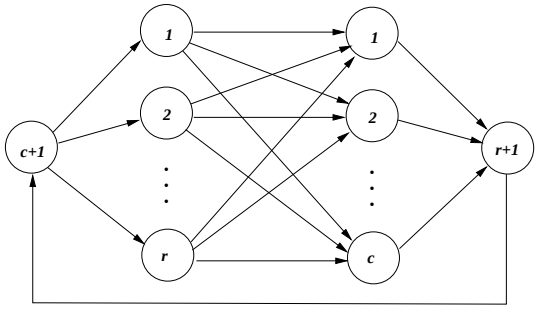

Fig. 6 Network representing constraints (1).

$$\begin{aligned}
\sum_{i=1}^{r} a_{ijk} &= a_{(r+1)jk} \quad j = 1 \ldots c, \;\; k = 1 \ldots l \\
\sum_{j=1}^{c} a_{ijk} &= a_{i(c+1)k} \quad i = 1 \ldots r, \;\; k = 1 \ldots l \\
\sum_{k=1}^{l} a_{ijk} &= a_{ij(l+1)} \quad i = 1 \ldots r, \;\; j = 1 \ldots c.
\end{aligned} \tag{2}$$

Note the above constraints correspond to a *cube* of data. Rearranging (2), these constraints can be modeled as a multicommodity network [5]. Variables $x_{ijk}, i = 1,\ldots,r, j = 1,\ldots,c, k = 1,\ldots,l$ are ordered according to k, i.e., $x = (x_{ij1}^T,\ldots,x_{ijl}^T)^T$. Each group for a particular k contains rc variables, and it corresponds to a layer of the cube of data. Each layer is a two-dimensional table, which is modeled as the network of Figure 6. Data for each particular layer (or level) corresponds to a commodity. The l commodities are linked by capacity constraints, forcing that the sum for all the commodities (levels) is equal to the marginal level. The resulting constraint matrix structure is

$$A = \begin{array}{c} \begin{array}{cccc} a_{ij1} & a_{ij2} & \ldots & a_{ijl} \end{array} \\ \left[\begin{array}{cccc} N & & & \\ & N & & \\ & & \ddots & \\ & & & N \\ I & I & \ldots & I \end{array}\right] \end{array} \begin{array}{l} \text{for } k = 1 \\ \text{for } k = 2 \\ \vdots \\ \text{for } k = l \\ \text{linking constraints,} \end{array} \tag{3}$$

N being the node-arc incidence network matrix for the two-dimensional tables of each level, and $I \in \mathbb{R}^{rc \times rc}$ being the identity matrix. Exploiting this structure, significant computational savings can be obtained [7, 10].

2.2.3 Hierarchical tables

In general, hierarchical tables have to be modeled as a general linked table. However, for the particular case of 1H2D tables, as that of Figure 4, it is possible to obtain a

network representation. In short, the algorithm for building the network of a 1H2D table consists of the following stages [9]:

1. Build a tree of subtables representing the structure of the 1H2D (i.e., for table of Figure 4, the root node would be the left table; the middle table would be a descendant of the root table; and the right table would be a descendant of the middle table).
2. Search all the subtables of the tree using for instance a breadth-first-search, and build the breadth-first-list.
3. Build the networks for each subtable.
4. For all the subtables in the breadth-first-list, embed the network of a table within the table of its parent table.

The above procedure is done in linear time. For instance, for the 1H2D table of Figure 4 after the first iteration we would get the network of Figure 7; after the second and last iteration the definitive network of Figure 8 would be obtained. This network model was successfully used for a fast heuristic for protection of 1H2D tables in [9].

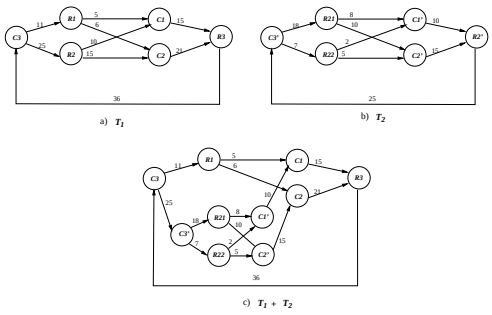

Fig. 7 Intermediate network representing 1H2D table of Figure 4 (first iteration).

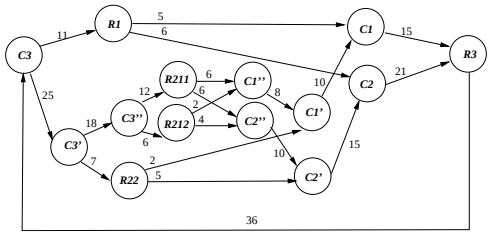

Fig. 8 Final network representing 1H2D table of Figure 4 (second iteration).

2.2.4 Linked tables

Any table can be modeled as a set of n cells $a_i, i = 1,\ldots,n$, which satisfy a set of m linear relations $Aa = b$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. If the table is positive then we may add bounds $a_i \geq 0, i = 1,\ldots,n$. Each row of matrix $A = (a_{ij}), i = 1,\ldots,m, j = 1,\ldots,n$ is related to a table linear relation, and $a_{ij} \in \{1,0,-1\}$. The value -1 of every equation is related to the marginal cell. The tables of above subsubsections are particular cases where A is either a node-arc incidence network flows matrix, or a multicommodity network flows matrix. In real world problems the dimension of n and m can be very large, up to millions of cells. Some huge instances can be found in `http://www-eio.upc.es/~jcastro/data.html`.

3 Sensitive cells and sensitivity rules

Sensitivity rules are used for detection of the set of cells with disclosure risk. For frequency tables, the *threshold value* rule is mostly used. For magnitude tables, both the (n,k) and the p% can be used, the latter being in general preferred. The three rules are outlined below. More practical details can be found in [28].

3.1 The threshold rule for frequency tables

In a frequency table, a cell is considered sensitive (i.e., its value has to be protected) if less than t respondents contribute to this cell. An usual value could be $t = 3$. Although this rule could also be applied to magnitude tables, this is not a good practice, since it misses the contribution of each respondent to the cell value.

3.2 The (n,k) and p% rules for magnitude tables

The (n,k) rule (also named *dominance rule*) considers a cell is sensitive if n or less respondents contribute to a k% (or more) of the cell value. For instance, for a cell $100 = 30+30+20+10+10$ (i.e., cell of value 100 and 5 respondents with contributions 30, 30, 20, 10, 10), if $n = 1$ and $k = 50$ then the cell is non-sensitive: any respondent contribution is less than a 50% of the cell value; however, if $n = 2$ and $k = 50$ then the cell is sensitive since $30+30 > 100\cdot 0.5$. The (n,k) rule tries to avoid that a coalition of n respondents could obtain accurate estimates of the other respondents contributions. Some usual values are, e.g., $(n = 3, k = 75)$.

For the p% rule a cell is sensitive if some respondent may obtain an estimate of other respondent contribution within a p% precision. The worse case—the one considered in practice–is obtained when the respondent with the second largest con-

tribution tries to estimate the value of the respondent with the highest contribution. For instance, for the cell $100 = 55+30+10+3+2$ (i.e., cell of value 100 and 5 respondents with contributions 55, 30, 10, 3, 2), the second respondent knows that the value of the first respondent is at most $100-30=70$; the estimate of the first respondent done by the second is 70. If $p = 20\%$, since $70 > (1+20/100)\cdot 55 = 66$, then the cell in non-sensitive. If $p = 30$, since $70 < (1+30/100)\cdot 55 = 71.5$, the cell is considered sensitive. In general, for a cell $X = x_1+x_2+\cdots+x_t$ with t respondents and $x_1 \geq x_2 \geq \cdots \geq x_t$, the estimate of x_1 is $\hat{x}_1 = X - x_2$, and the cell is sensitive if

$$\hat{x}_1 - x_1 < p/100 x_1 \Leftrightarrow X - x_1 - x_2 < p/100 x_1. \tag{4}$$

In general, the p% is preferred to the (n,k) rule. Indeed the (n,k) may wrongly consider as non-sensitive sensitive cells and vice-versa. The following example, from [30], illustrates this situation. Consider the rule $(n=1,k=60)$. When applied to the cell $100 = 59+40+1$, this is considered non-sensitive, since $59 < 0.6\cdot 100$. On the other hand, the cell $100 = 61+20+19$ is considered sensitive, since $61 > 0.6\cdot 100$. However, for the cell declared non-sensitive, the second respondent gets a too tight estimation of the first one of value 60: $100-59=61$. Similarly, for the cell considered sensitive, the estimation by second respondent would be $100-20=80$, far from the real value.

Situations as those of the above paragraph could be avoided by using a rule $(n=2,k)$, but even in this case the p% rule is preferred. This is shown by noting that the $(n=2,k)$ rule considers a cell as sensitive if

$$x_1+x_2 > k/100 X \Leftrightarrow X - x_2 - x_1 < (1-k/100)X. \tag{5}$$

Comparing (4) and (5), it is seen that in both cases a cell is sensitive if $(X-x_2)-x_1$, i.e. the difference between the estimation of x_1 made by second respondent and x_1, is less than a certain percentage of either the first respondent value x_1 in (4) or the cell value X in (5). Note that the p% rule is more natural, and that the $(n=2,k)$ suffers from overprotection. Indeed, for some particular values of p and k it can be proved that the set of sensitive cells provided by the rule p% is a subset of the set obtained with $(n=2,k)$ This is clearly seen in the following result [28].

Proposition 1. *For p and k such that $k = 100\frac{100}{100+p}$, every non-sensitive cell for the rule $(n=2,k)$ is also a non-sensitive cell for the rule p%; but the reverse implication does not hold.*

Proof. First we prove the direct implication. If a cell $X = x_1+x_2+\cdots+x_t$ is non-sensitive for $(n=2,k)$ then by (5)

$$x_1+x_2 \leq \frac{k}{100}X = \frac{100}{100+p}X \Rightarrow (X-x_2)-x_1 \geq \left(1-\frac{100}{100+p}\right)X = \frac{p}{100+p}X, \tag{6}$$

and also

$$x_1 \leq \frac{k}{100}X = \frac{100}{100+p}X \Rightarrow \frac{p}{100}x_1 \leq \frac{p}{100+p}X. \tag{7}$$

Connecting inequalities (6)–(7) we have

$$(X - x_2) - x_1 \geq \frac{p}{100+p} X \geq \frac{p}{100} x_1,$$

thus the cell is non-sensitive for the rule p%.

To show that the reverse implication is not true, we consider a counterexample. For $p = 10\%$ and $k = 100 \cdot 100/(100+p)$, a cell $X = 110$ with $x_1 = 52$, $x_2 = 50$ is non-sensitive for the p% rule, since $\hat{x}_1 - x_1 = (110-50) - 52 = 60 - 52 > 52 \cdot p/100$. However it is sensitive for the $(n = 2, k)$ rule, because $x_1 + x_2 = 102 > k/100 \cdot 110 = 100$. □

4 Tabular data protection methods

Tabular data protection methods can be classified as

- Non-perturbative: they don't change the original data, instead they "hide" data or change the table structure. Among them we find *recoding* and *cell suppression*.
- Perturbative: they provide an alternative table with modified values. *Controlled rounding* and *controlled tabular adjustment* belong to this class.

The above four methods are introduced and outlined below. References for a full description of the solution approaches can be found within each subsection.

4.1 Recoding

This simple procedure consists in aggregating or changing some of the categorical variables that define the table, in order to satisfy sensitivity rules. This is shown in the example of Figure 9, whose tables report the number of respondents for "profession" and "municipality". This method is implemented in the τ-Argus software [27]. The main advantages of this approach are its simplicity and that it works fine in practice. The main inconvenience is that it changes the table structure; an excessive aggregation may significantly reduce the utility of the resulting table.

Original table

	P_1	P_2	P_3	P_4	P_5	TOTAL
M_1	20	15	30	20	10	95
M_2	72	20	**1**	30	10	133
M_3	38	38	15	40	11	142
TOTAL	130	73	46	90	31	370

Recoded table

	P_1	P_2+P_3	P_4	P_5	TOTAL
M_1	20	45	20	10	95
M_2	72	**21**	30	10	133
M_3	38	53	40	11	142
TOTAL	130	119	90	31	370

Fig. 9 Original and recoded table after aggregation of professions P_2 and P_3.

4.2 Cell suppression

Given a set of sensitive cells to be protected (named *primary* cells), the cell suppression method removes them, and an additional set of cells named *secondary* cells to guarantee that the value of primary cells can not be disclosed. The purpose of the *cell suppression problem* (CSP) is to find the set of secondary cells that minimize some information loss criteria. Figure 10 shows an example of a two-dimensional table with only one primary cell in boldface; removing this cell is not enough, since its value can be retrieved from marginals, thus forcing the suppression of three additional complementary cells.

Fig. 10 Original table with primary cell in boldface, and protected table after suppression of three secondary cells.

Original table

	P_1	P_2	P_3	TOTAL
M_1	20	24	28	72
M_2	38	38	**40**	116
M_3	40	39	42	121
TOTAL	98	101	110	309

Protected table

	P_1	P_2	P_3	TOTAL
M_1		24		72
M_2		38		116
M_3	40	39	42	121
TOTAL	98	101	110	309

From the protected table of Figure 10, any attacker may deduce a lower and upper bound for the primary cell. Indeed, considering variables x_{11}, x_{13}, x_{21}, x_{23} for the primary and secondary cells, a lower bound $\underline{a_{23}}$ and an upper bound $\overline{a_{23}}$ for the primary cell can be obtained by solving

$$\begin{array}{l} \underline{a_{23}} = \min x_{23} \\ \text{subjectto } x_{11}+x_{13} = 72-24 \\ \qquad\quad x_{21}+x_{23} = 116-38 \\ \qquad\quad x_{11}+x_{21} = 98-40 \\ \qquad\quad x_{13}+x_{23} = 110-42 \\ \qquad\quad (x_{11},x_{13},x_{21},x_{23}) \geq 0 \end{array} \quad \text{and} \quad \begin{array}{l} \overline{a_{23}} = \max x_{23} \\ \text{subjectto } x_{11}+x_{13} = 72-24 \\ \qquad\quad x_{21}+x_{23} = 116-38 \\ \qquad\quad x_{11}+x_{21} = 98-40 \\ \qquad\quad x_{13}+x_{23} = 110-42 \\ \qquad\quad (x_{11},x_{13},x_{21},x_{23}) \geq 0. \end{array} \tag{8}$$

The solutions to (8) are $\underline{a_{23}} = 20$ and $\overline{a_{23}} = 68$. If, for instance, *lower* and *upper protection levels* of $lpl = upl = 10$ were imposed (i.e., the protection pattern must guarantee that no attacker can deduce a value of the sensitive cell within the range $[40-lpl, 40+upl] = [30,50]$), then this cell would be protected by this suppression pattern since $\underline{a_{23}} = 20 < 30$ and $\overline{a_{23}} = 68 > 50$.

The above example illustrated the basics of CSP. A general formulation is now provided. Any instance of CSP is defined by the following parameters:

- A general table $a_i, i = 1,\ldots,n$, with m linear relations $Aa = b$, $a = (a_1,\ldots,a_n)^T$ being the vector of cell values.
- Upper and lower bounds u and l for the cell values, which are assumed to be known by any attacker: $l \leq a \leq u$ (e.g., $l = 0$, $u = +\infty$ for a positive table).
- Vector of nonnegative weights associated to the cell suppressions $w_i, i = 1,\ldots,n$. If $w_i = 1$ the number of cells is minimized; if $w_i = a_i$ the value suppressed is minimized.

- Set $\mathscr{P} \subseteq \{1,\ldots,n\}$ of primary or sensitive cells.
- Lower and upper protection levels for each primary cell lpl_p and upl_p $p \in \mathscr{P}$ (usually either a fraction of a_p or obtained from the sensitivity rules p% and (n,k)).

CSP looks for a set $\mathscr{S}$ of secondary cells to be removed such that for all $p \in \mathscr{P}$

$$\underline{a_p} \leq a_p - lpl_p \quad \text{and} \quad \overline{a_p} \geq a_p + upl_p, \tag{9}$$

$\underline{a_p}$ and $\overline{a_p}$ being defined as

$$\underline{a_p} = \begin{array}{ll} \min & x_p \\ \text{subject to} & Ax = b \\ & l_i \leq x_i \leq u_i \;\; i \in \mathscr{P} \cup \mathscr{S} \\ & x_i = a_i \;\; i \notin \mathscr{P} \cup \mathscr{S} \end{array} \quad \text{and} \quad \overline{a_p} = \begin{array}{ll} \max & x_p \\ \text{subject to} & Ax = b \\ & l_i \leq x_i \leq u_i \;\; i \in \mathscr{P} \cup \mathscr{S} \\ & x_i = a_i \;\; i \notin \mathscr{P} \cup \mathscr{S}. \end{array} \tag{10}$$

The classical model for CSP was originally formulated in [29]. It considers two sets of variables

- $y_i \in \{0,1\}, i = 1,\ldots,n$ is 1 if cell has to be suppressed, 0 otherwise.
- Fore each primary cell $p \in \mathscr{P}$, two auxiliary vectors $x^{l,p} \in \mathbb{R}^n$ and $x^{u,p} \in \mathbb{R}^n$, which represent cell deviations (positive or negative) from the original a_i values; they are needed to guarantee the protection levels.

The resulting model is

$$\begin{array}{ll} \min & \displaystyle\sum_{i=1}^{n} w_i y_i \\ \text{subject to} & \left.\begin{array}{rl} & Ax^{l,p} = 0 \\ (l_i - a_i)y_i \leq & x_i^{l,p} \leq (u_i - a_i)y_i \quad i = 1,\ldots,n \\ & x_p^{l,p} \leq -lpl_p \end{array}\right\} \\ & \left.\begin{array}{rl} & Ax^{u,p} = 0 \\ (l_i - a_i)y_i \leq & x_i^{u,p} \leq (u_i - a_i)y_i \quad i = 1,\ldots,n \\ & x_p^{u,p} \geq upl_p \end{array}\right\} \quad \forall\, p \in \mathscr{P} \\ & y_i \in \{0,1\} \quad i = 1,\ldots,n. \end{array} \tag{11}$$

The inequality constraints of (11) with both right- and left-hand sides impose bounds on $x_i^{l,p}$ and $x_i^{u,p}$ when $y_i = 1$, and prevent deviations in non-suppressed cells (i.e., $y_i = 0$). Clearly, the constraints of (11) guarantee that the solutions of the linear programs (10) will satisfy (9).

Model (11) is the basis of several solution methods, either optimal or heuristic. Note however that it can not be used directly as formulated here, because (11) gives rise to a MILP problem of n binary variables, $2n|\mathscr{P}|$ continuous variables, and $2(m+2n)|\mathscr{P}|$ constraints. This problem is very large even for tables of moderate size and number of primary cells. For instance, for a table of 8000 cells, 800

primaries, and 4000 linear relations, we obtain a MILP with 8000 binary variables, 12,800,000 continuous variables, and 32,000,000 constraints.

The unique currently optimal solution approach decomposes (11) by means of a Benders decomposition [3]. Initially applied to two-dimensional tables [23], it was later extended to general tables [24]. The main benefit of this approach is that it guarantees an optimal solution. The main drawback is that the number of cuts needed (i.e., iterations of Benders method) may be very large, resulting in a prohibitive computational time. This does not happen for two-dimensional tables (the approach is very fast for this kind of tables), but it becomes computationally very expensive for more complex tables, as it will be shown below in a numerical example. This method is implemented in the τ-Argus package [27].

Most heuristic approaches for (11) find a feasible, hopefully good point, by network optimization algorithms (in particular, minimum-cost network flows, and shortest paths [1]). Unfortunately, those heuristics can only be used in tables that accept a network representation: two-dimensional and 1H2D hierarchical tables (the latter is however an interesting case for NSAs). Some attempts have been made for extending them to three-dimensional tables [18], but as mentioned in Section 2.2.2, three-dimensional tables correspond to multicommodity flows, and therefore "standard single-commodity" network optimization procedures are not valid (and rather unsuccessful). Among those heuristics we find the seminal paper [29], and [5, 14], which rely on minimum-network cost flows. For general tables [4] suggested an efficient procedure based on shortest paths. Some of those ideas were sensibly combined in the approach of [9], based on shortest paths but valid for positive tables. This approach is very efficient, but it can only be applied to either two-dimensional or 1H2D hierarchical tables. This method is implemented in the τ-Argus package.

We finally mention two other heuristics, which are also available in the τ-Argus package. The *hypercube* [25], initially developed for k-dimensional tables, is a simple and fast procedure. For two-dimensional tables it can be seen as a network flows approach that only considers a subset of the flows (thus providing less quality solutions than heuristics based on network optimization). Although it is efficient, in practice tends to oversuppress cells and, moreover, it does not guarantee a feasible solution (indeed, it finishes with some underprotected cells). Some of the above drawbacks are also shared by the other heuristic, named *Hitas* [19]. That approach decomposes any table in a tree of smaller two-dimensional subtables and locally protects them by the previously cited optimal Benders decomposition approach. Since some linking constraints between subtables are removed, the final solution is not guaranteed to be feasible. However, the quality of the solutions is in general acceptable.

It is not easy to compare the above procedures computationally, since the source code is not available. However, they can be run with the same table from the τ-Argus package, which implements four of them: the optimal approach of [24], the shortest paths heuristic of [9], and the two (infeasible) heuristics of [19] and [25]. To compare them, in [26] a toy table 1H2D was generated with τ-Argus. This table was obtained from the microdata file accompanying the τ-Argus distribution, crossing categorical variables "industry code" and "size", and using "var2" as explanatory

variable. The results are reported in Table 1. Columns "#supp." and "#val. supp." provide information about the solution reported (number of suppressions, and total value suppressed, respectively). The total value suppressed is the objective function to be minimized. Column "CPU sec" provides the CPU time. Time limits of 2 and 10 minutes were set for the optimal procedure. Even such a small instance is very difficult for the Benders decomposition approach, but it provides a better objective. The shortest paths heuristic provides better results than the other heuristics (and it is guaranteed to provide a feasible solution). In addition it requires less than 1% of the CPU time of the optimal approach for a solution with an objective value only a 20% worse. However, if the table was more complex (instead of 1H2D) the shortest paths heuristic could not be used.

Table 1 Results for table "IndustryCode $\times$ Size $\rightarrow$ Var2", from microdata file of τ-Argus distribution.

Method	#supp.	#val. supp.	CPU sec†
Hypercube	637	15494253	9
HiTas	528	9016562	15
Shortest paths	538	8795130	4
Benders decomposition	557	7830730	120*
Benders decomposition	483	7216286	622*

† Results on a PC with one AMD Athlon 44 00+ 64 bits dual core
* Time limit

4.3 Controlled rounding

The method of *rounding* achieves protection by rounding all cell tables to a multiple of a certain base number r. Figure 11 shows an example of a two-dimensional table using a base number $r = 5$. Note that the total cell could not be rounded to the closest multiple of 5, otherwise the resulting table would not be additive. This variant that guarantees additivity is named *controlled rounding*, instead of rounding.

Fig. 11 Original and rounded table using a base number $r = 5$.

Original table

	P_1	P_2	P_3	TOTAL
M_1	20	24	28	72
M_2	38	38	40	116
M_3	40	39	42	121
TOTAL	98	101	110	**309**

Rounded table

	P_1	P_2	P_3	TOTAL
M_1	20	25	30	75
M_2	40	40	40	120
M_3	40	40	40	120
TOTAL	100	105	110	**315**

Although controlled rounding was already in use two decades ago [15], some recent extensions using lower and upper protection levels have been considered [31]. The complexity of the resulting model is similar to that of cell suppression, resulting

in a large MILP which is solved by Benders decomposition [3]. This model is implemented in the τ-Argus package. One of the main drawbacks of controlled rounding is that it forces deviations for all the cells that are not originally a multiple of the base r, reducing the utility of the resulting table. In addition, to guarantee additivity, total cells have also to be rounded, likely to a multiple which can be far from the original value. The method of next subsection, which also perturbs cell values, avoids some of these inconveniences of controlled rounding.

4.4 Controlled tabular adjustment

Given a table, a set of sensitive cells, and some lower and upper protection levels, the purpose of *controlled tabular adjustment* (also known as *minimum-distance controlled tabular adjustment* or simply *CTA*) is to find the closest safe table to the original one (i.e., the closest table that meets the protection levels). Figure 12 shows an example for a small two-dimensional table with one sensitive cell in boldface, with lower and upper protection levels equal to five (left table of the Figure). If the protection sense is "lower", then the value published for the sensitive cell should be less or equal than 35; the optimal adjusted table for this case is shown in the middle table of Figure 12. If the protection sense is "upper", then the value must be greater or equal than 45, as shown in the right table of Figure 12.

Original table

	P_1	P_2	P_3	TOTAL
M_1	20	24	28	72
M_2	38	38	**40**	116
M_3	40	39	42	121
TOTAL	98	101	110	309

Adjusted table, lower protection sense

	P_1	P_2	P_3	TOTAL
M_1	15	24	33	72
M_2	43	38	**35**	116
M_3	40	39	42	121
TOTAL	98	101	110	309

Adjusted table, upper protection sense

	P_1	P_2	P_3	TOTAL
M_1	25	24	23	72
M_2	33	38	**45**	116
M_3	40	39	42	121
TOTAL	98	101	110	309

Fig. 12 Original table with sensitive cell in boldface, of lower and upper protection levels equal to five. Protected tables with "lower protection sense" and "upper protection sense" (i.e., value of sensitive is respectively reduced and increased by five units).

CTA was introduced in the manuscript [17] and, independently and in an extended form, in [8] (in the latter they were named minimum-distance controlled perturbation methods). The parameters that define any CTA instance are the same than for the cell suppression problem (see, Subsection 4.2), i.e.:

- A general table $a_i, i = 1, \ldots, n$, with m linear relations $Aa = b$.
- Upper and lower bounds u and l for the cell values, assumed to be known by any attacker: $l \leq a \leq u$.
- Vector of nonnegative weights associated to the cell perturbations $w_i, i = 1, \ldots, n$.
- Set $\mathscr{P} \subseteq \{1, \ldots, n\}$ of sensitive cells.
- Lower and upper protection levels for each primary cell lpl_p and upl_p $p \in \mathscr{P}$.

CTA finds the safe table x closest to a, using some distance L_w:

$$\begin{aligned} &\min_x \ ||x-a||_{L(w)} \\ &\text{subject to } Ax = b \\ &\qquad l_x \le x \le u_x \\ &\qquad x_p \le a_p - lpl_p \text{ or } x_p \ge a_p + upl_p \quad p \in \mathscr{P}. \end{aligned} \tag{12}$$

Defining $z = x - a$, $l_z = l_x - a$ and $u_z = u_x - a$, (12) can be recast in terms of deviations:

$$\begin{aligned} &\min_z \ ||z||_{L(w)} \\ &\text{subject to } Az = 0 \\ &\qquad l_z \le z \le u_z \\ &\qquad z_p \le -lpl_p \text{ or } z_p \ge upl_p \quad p \in \mathscr{P}. \end{aligned} \tag{13}$$

To model the "or" constraints it is necessary to consider binary variables $y_p \in \{0,1\}$, $p \in \mathscr{P}$, such that $y_p = 1$ if cell is "upper protected" (i.e, $z_p \ge upl_p$), and $y_p = 0$ if it is "lower protected" ($z_p \le -lpl_p$). For the particular case of distance L_1, it is also needed a pair of variables z_i^+ and z_i^-, such that $z_i = z_i^+ - z_i^-$ and $|z_i| = z_i^+ + z_i^-$. The resulting MILP model is

$$\begin{aligned} &\min_{z^+,z^-} \ \sum_{i=1}^{n} w_i (z_i^+ + z_i^-) \\ &\text{subject to } A(z^+ - z^-) = 0 \\ &\qquad 0 \le z_i^+ \le u_{z_i} \quad i \notin \mathscr{P} \\ &\qquad 0 \le z_i^- \le -l_{z_i} \quad i \notin \mathscr{P} \\ &\qquad upl_i y_i \le z_i^+ \le u_{z_i} y_i \quad i \in \mathscr{P} \\ &\qquad lpl_i (1-y_i) \le z_i^- \le -l_{z_i}(1-y_i) \quad i \in \mathscr{P} \\ &\qquad y_i \in \{0,1\} \quad i \in \mathscr{P}. \end{aligned} \tag{14}$$

Problem (14) has $|\mathscr{P}|$ binary variables, $2n$ continuous variables and $m + 4|\mathscr{P}|$ constraints. The size of (14) is much less than that of the the cell suppression formulation (11). For instance, for a table of 8000 cells, 800 primaries, and 4000 linear relations, CTA formulates a MILP of 800 binary variables, 16000 continuous variables and 7200 constraints (these figures were 8000, 12,800,000 and 32,000,000 for CSP).

Because of the smaller size of CTA compared to other approaches, it is possible to apply state-of-the-art MILP solvers. Such an implementation was developed using both CPLEX and XPRESS in a package to be used by Eurostat [13]. However, real world large and complex instances are still difficult for such generic solvers, and some preliminary work has been started using optimal approaches based on Benders reformulation [11], and heuristics [26]. The benefits of CTA are not limited to a smaller size of the resulting MILP problem. CTA can be easily extended with constraints to meet some data quality criteria [16]. It has also been experimentally observed that the quality of CTA solution is comparable (in some instances even better) than that of CSP [12]: indeed the number of cells with significantly large deviations is much smaller than the number of cells removed by CSP.

5 Conclusions

This chapter introduced some of the currently most used techniques for tabular data protection. All of them share, at different degrees, the same computational drawbacks: they result in large difficult MILP optimization problems. Current research for improving the solution of these MILP problems is being undertaken, mainly for the most recent method, controlled tabular adjustment. That research makes use of recent advances in mathematical optimization. There are alternative protection methods, like *interval protection* or *partial cell suppression* which result in a very large, even massive, linear programming problem. Some approaches based on Benders decomposition were suggested in the literature. Being a continuous optimization problem, specialized interior-point methods for structured problems can also be a very efficient alternative. This is research to be conducted in the near future in this challenging field.

Acknowledgements This work has been supported by grant MTM2006-05550 of the Spanish Ministry of Science and Education.

References

1. R.K. Ahuja, T.L. Magnanti and J.B. Orlin (1993), *Network flows. Theory, Algorithms and Applications*, Prentice Hall, Upper Saddle River, NJ.
2. M. Bacharach (1966), Matrix rounding problems, *Management Science*, 9, 732-742.
3. J.F. Benders (2005), Partitioning procedures for solving mixed-variables programming problems, *Computational Management Science*, 2 3–19. English translation of the original paper appeared in *Numerische Mathematik*, 4 238–252 (1962).
4. F.D. Carvalho, N.P. Dellaert and M.D. Osório (1994), Statistical disclosure in two-dimensional tables: general tables, *Journal of the American Statistical Association*, 89 1547–1557.
5. J. Castro (2002), Network flows heuristics for complementary cell suppression: an empirical evaluation and extensions, *Lecture Notes in Computer Science*, 2316 59–73.
6. J. Castro (2004), A fast network flows heuristic for cell suppression in positive tables, *Lecture Notes in Computer Science*, 3050 136-148.
7. J. Castro (2005), Quadratic interior-point methods in statistical disclosure control, *Computational Management Science*, 2(2) 107-121.
8. J. Castro (2006), Minimum-distance controlled perturbation methods for large-scale tabular data protection, *European Journal of Operational Research*, 171 39-52
9. J. Castro (2007), A shortest-paths heuristic for statistical data protection in positive tables, *INFORMS Journal on Computing*, 19(4) 520-533.
10. J. Castro (2007), An interior-point approach for primal block-angular problems, *Computational Optimization and Applications*, 36 195–219.
11. J. Castro and D. Baena (2008). Using a mathematical programming modeling language for optimal CTA, *Lecture Notes in Computer Science*, 5262 1–12.
12. J. Castro and S. Giessing (2006), Testing variants of minimum distance controlled tabular adjustment, in *Monographs of Official Statistics. Work session on Statistical Data Confidentiality*, Eurostat-Office for Official Publications of the European Communities, Luxembourg, 2006, 333-343. ISBN 92-79-01108-1

13. J. Castro, A. González and D. Baena (2009), 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.
14. L.H. Cox (1995), Network models for complementary cell suppression, *Journal of the American Statistical Association*, 90 1453–1462.
15. L.H. Cox and J.A. George (1989), Controlled rounding for tables with subtotals, *Annals of Operations Research*, 20 141–157.
16. L.H. Cox, J.P. Kelly and R. Patil (2005), Computational aspects of controlled tabular adjustment: algorithm and analysis. B. Golden, S. Raghavan, E. Wassil, eds. *The Next wave in Computer, Optimization and Decision Technologies*, Kluwer, Boston, MA, 45–59.
17. R.A. Dandekar and L.H. Cox (2002), Synthetic tabular data: An alternative to complementary cell suppression, manuscript, Energy Information Administration, US Department of. Energy.
18. N.P. Dellaert and W.A. Luijten (1999), Statistical disclosure in general three-dimensional tables, *Statistica Neerlandica*, 53 197–221.
19. P.P. de Wolf (2002), HiTaS: A heuristic approach to cell suppression in hierarchical tables, *Lecture Notes in Computer Science* 2316 74–82.
20. J. Domingo-Ferrer and L. Franconi (eds.) (2006), *Lecture Notes in Computer Science. Privacy in Statistical Databases* 4302, Springer, Berlin.
21. J. Domingo-Ferrer and Y. Saigin (eds.) (2008), *Lecture Notes in Computer Science. Privacy in Statistical Databases* 5262, Springer, Berlin.
22. J. Domingo-Ferrer and V. Torra (eds.) (2004), *Lecture Notes in Computer Science. Privacy in Statistical Databases* 3050, Springer, Berlin.
23. M. Fischetti and J.J. Salazar-González (1999), Models and algorithms for the 2-dimensional cell suppression problem in statistical disclosure control. *Mathematical Programming*, 84 283–312.
24. M. Fischetti and J.J. Salazar-González (2001), Solving the cell suppression problem on tabular data with linear constraints, *Management Science* 47 1008–1026.
25. S. Giessing and D. Repsilber (2002), Tools and strategies to protect multiple tables with the GHQUAR cell suppression engine, *Lecture Notes in Computer Science* 2316 181–192.
26. A. González, J. Castro (2009), Block coordinate descent decomposition for statistical data protection using controlled tabular adjustment, Research Report DR 2009/10, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, 2009. Submitted.
27. A. Hundepool, A. van de Wetering, R. Ramaswamy, P.P. de Wolf, S. Giessing, M. Fischetti, J.J. Salazar-González, J. Castro, P. Lowthian (2007), *τ-Argus User's Manual*.
28. A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, R. Lenz, J. Longhurst, E. Schulte-Nordholt, Giovanni Seri, P.P. de Wolf (2007), *Handbook on Statistical Disclosure Control*, CENEX SDC. Available on-line at `http://neon.vb.cbs.nl/casc/.\ SDC_Handbook.pdf`.
29. J.P. Kelly, B.L. Golden and A.A. Assad (1992), Cell suppression: disclosure protection for sensitive tabular data. *Networks*, 22 28–55.
30. D.A. Robertson and R. Ethier (2002), Cell suppression: experience and theory , *Lecture Notes in Computer Science*, 2316 8–20.
31. J.J. Salazar-González (2006), Controlled rounding and cell perturbation: statistical disclosure limitation methods for tabular data, *Mathematical Programming* 105 583–603.
32. J.J. Salazar-González (2008), Statistical confidentiality: Optimization techniques to protect tables, *Computers and Operations Research* 35 1638–1651.
33. L. Willenborg and T. de Waal (eds.) (2000) *Lecture Notes in Statistics. Elements of Statistical Disclosure Control* 155, Springer, New York.