

Summer 2017

# Optimization Methods for Tabular Data Protection

Iryna Petrenko

OPTIMIZATION METHODS FOR TABULAR DATA PROTECTION

by

IRYNA PETRENKO

(Under the Direction of Goran Lesaja)

ABSTRACT

In this thesis we consider a minimum distance Controlled Tabular Adjustment (CTA) model for statistical disclosure limitation (control) of tabular data. The goal of the CTA model is to find the closest safe table to some original tabular data set that contains sensitive information. The measure of closeness is usually measured using $\ell_1$ or $\ell_2$ norm; with each measure having its advantages and disadvantages. According to the given norm CTA can be formulated as an optimization problem: Liner Programing (LP) for $\ell_1$, Quadratic Programing (QP) for $\ell_2$. In this thesis we present an alternative reformulation of $\ell_1$-CTA as Second-Order Cone (SOC) optimization problems. All three models can be solved using appropriate versions of Interior-Point Methods (IPM). The validity of the new approach was tested on the randomly generated two-dimensional tabular data sets. It was shown numerically, that SOC formulation compares favorably to QP and LP formulations.

INDEX WORDS: Statistical disclosure limitation, Controlled tabular adjustment, Liner programing optimization problem, Quadratic programing optimization problem, Symetric cone, Conic optimization problem, Interior point method

2009 Mathematics Subject Classification: Computational Science

# OPTIMIZATION METHODS FOR TABULAR DATA PROTECTION

by

IRYNA PETRENKO

B.S., Odesa National Academy of Telecommunication, Ukraine, 2007

M.S., Odesa National Academy of Telecommunication, Ukraine, 2008

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

STATESBORO, GEORGIA

# OPTIMIZATION METHODS FOR TABULAR DATA PROTECTION

by

IRYNA PETRENKO

| | |
|---|---|
| Major Professor: | Goran Lesaja |
| Committee: | Emil Iacob |
| | Scott Kersey |

Electronic Version Approved:
July 2017

TABLE OF CONTENTS

Page

# LIST OF TABLES

Table Page

# LIST OF FIGURES

Figure Page

# LIST OF SYMBOLS

| | |
|---|---|
| $\mathbb{R}$ | Real Numbers |
| $\mathbb{C}$ | Real Numbers |
| $\mathbb{Z}$ | Integers |
| $\mathbb{N}$ | Natural Numbers |
| $\mathbb{N}_0$ | Natural Numbers including 0 |
| $L_p(\mathbb{R})$ | $p$-integrable functions over $\mathbb{R}$ |
| $L(X, Y)$ | Linear maps from $X$ to $Y$ |
| rank$(T)$ | Rank of a linear map |

# CHAPTER 1

# INTRODUCTION

Today people are faced with task of processing huge volume of information. The data can come from any spheres of human life beginning from some personal information and ending by different branches of industry and business. There is often a need to protect the confidentiality of the informations provided by the respondents before realizing data to the public .

To describe the term 'protecting confidentiality' means, we need to provide some definitions. This usually involves the notion of a hypothetical *intruder* who might break this confidentiality. There are three key parties: (i) the *respondent* who provides the data, (ii) the *agency* (such that U.S. Census Bureau,The Federal Committeeon Statistical Methodology, The American Statistical Association, The European Union SDC group, etc.) which collects the data, releases statistical outputs and designs the Statistical Disclosure Control strategy, and (iii) the hypothetical *intruder* who has access to these outputs and seeks to use them to disclose information about the respondent.

*Statistical Disclosure Control (SDC)* or *Statistical Disclosure Limitation (SDL)* seeks to protect statistical data in such a way that they can be released without giving away confidential information that can be linked to specific individuals or entities.

SDL techniques can be defined as the set of methods to reduce the risk of disclosing information on individuals, businesses or other organizations. SDL methods minimize the risk of disclosure to an acceptable level while releasing as much information as possible. There are two types of SDL methods: *perturbative* and *non-perturbative* methods. Perturbative methods falsify the data before publication by introducing an element of error purposely for confidentiality reasons. Non-perturbative methods reduce the amount of information released by suppression or aggregation of data.

The methodology of SDL corresponds to the type of initial data:

- *Tabular Data.*

  There are two types of tabular output:

  1. *Magnitude tables.* In a magnitude table, each cell value represents the sum of a particular response, across all respondents that belong to that cell.
  2. *Frequency tables.* In a frequency table, each cell value represents the number of respondents that fall into that cell.

- *Microdata*

  A microdata set $\mathbb{V}$ can be viewed as a file with $n$ records, where each record contains $m$ variables (also called attributes) on an individual respondent, who can be a person or an organization (e.g. a company). Microdata are the form from which all other data outputs are derived and they are the primary form that data are stored in.

  Depending on their sensitivity, the variables in an original unprotected microdata set can be classified into four categories which are not necessarily disjoint:

  1. *Identifiers.* These are variables that unambiguously identify the respondent.
  2. *Quasi-identifiers* or *key variables.* This is a set of variables in $\mathbb{V}$ that, in combination, can be linked with external information to re-identify (some of) the respondents to whom (some of) the records in $\mathbb{V}$ refer.
  3. *Confidential outcome variables.* These are variables which contain sensitive information on the respondent.
  4. *Non-confidential outcome variables.* Those variables which contain non-sensitive information on the respondent.

Depending on their data type, the variables in a microdata set can be classified as:

- *Continuous.* A variable is considered continuous if numerical and arithmetical operations can be performed on it.
- *Categorical.* A variable is considered categorical when it takes values over a finite set and standard arithmetical operations do not make sense. Two main types of categorical variables can be distinguished:
  - *Ordinal.* An ordinal variable takes values in an ordered range of categories. Thus, the $\leq$, max and min operators are meaningful with ordinal data.
  - *Nominal.* A nominal variable takes values in an unordered range of categories. The only possible operator is comparison for equality.

Since we defined SDL as a set of methods that reduce the disclosure risk, we defined *disclosure risk* as probability of disclosure with respect to specified sources of uncertainty. Or the term might be used loosely to emphasize not only the uncertainty about potential disclosure but also the potential harm that might arise from disclosure.

The SDL method will lead to some loss of information for the user of the statistical output. The term *utility* may be used to cover both the information provided by the statistical outputs and the quality of this information. Utility generally needs to be considered from the perspective of a *user* of the statistical outputs, who represents a key fourth party to add to the three parties referred to earlier: the respondent, the agency and the intruder.

The main challenge in SDL is how to deal with the trade-off between disclosure risk and utility. In general, the more the disclosure risk is reduced by an SDL method,

the lower will be the expected utility of the output. This trade-off may be formulated as an optimization problem.

Let $D$ - be the data that been provided by responder, $f(D)$ - be the statistical output, resulting from the use of an SDL method, $R[f(D)]$ - be a measure of the disclosure risk of the output, $U[f(D)]$ - be a measure of the utility of the output and $\epsilon$ - be a maximum tolerable risk that mentions by the agency. Then the basic challenge of SDL might be represented as the constrained optimization problem :

$$\text{for given } D \text{ and } \epsilon, \text{ find an SDL method } f(\cdot), \text{ which:}$$

$$\text{maximize } U[f(D)], \text{ subject to } R[f(D)] < \epsilon.$$

Monograph [1] provides the detailed description of all SDL methods and techniques.

# CHAPTER 2

# SDL METHODS OVERVIEW: MICRODATA

The main aim of SDL for microdata is to prevent confidential information from being linked to specific respondents when releasing a microdata file. We can describe that as, for given an original microdata set $\mathbb{V}$, the goal of SDL is to release a protected microdata set $\mathbb{V}'$ in such a way that: (i) disclosure risk (i.e. the risk that a user or an intruder can use $\mathbb{V}'$ to determine confidential variables on a specific individual among those in $\mathbb{V}$) is low; (ii) user analyses (regressions, means, etc.) on $\mathbb{V}'$ and $\mathbb{V}$ yield the same or at least similar results.

In terms of microdata the SDL process may be divided into the following steps:

(i) determination of the needs of confidentiality protection;

(ii) determination of the key characteristics and use of the data;

(iii) definition of the disclosure risk and assessment;

(iv) determination of the disclosure control methods;

(v) implementation.

## 2.1 Disclosure risk

Once the characteristics and use of the survey data have been clarified, it is time to start the proper analysis of the disclosure risk.

Intuitively, a unit is at risk of disclosure when it cannot be confused with several other units, i.e. if it can be singled out from the rest. Record-level definitions of risk, called individual risk, may be useful for two purposes. On the one hand, they can be exploited to protect data selectively, i.e. apply SDL procedures only to those records being at risk. On the other hand, they can be used to built an overall definition of

risk for the whole microdata file to be released, i.e. a global risk measure [2]. As the process to single a unit out of the others depends on the type of quasi-identifiers, also the definition of risk depends heavily on the type of the quasi-identifiers. We distinguish three cases:

1. All the quasi-identifiers are *categorical*.

2. The quasi-identifiers are *continuous*.

3. Some quasi-identifier are categorical and some are continuous.

The first case is the common situation found when microdata stem from social surveys, the other two cases are mainly present when dealing with business surveys. For the details see [1]

## 2.2 The SDL methods for Microdata

Choice of the most suitable SDL method is based on previous analyses regarding the type of users and the statistical methods applied to analyse the microdata, the objectives to be investigated, the constraints imposed by the production process and the policy of the agency.

Microdata protection methods can generate the protected microdata set $\mathbb{V}'$:

- either by *masking original data*, i.e. generating a modified version $\mathbb{V}'$of the original microdata set $\mathbb{V}$;

- or by *generating synthetic data* $\mathbb{V}'$ that preserve some statistical properties of the original data $\mathbb{V}$.

The masking methods divides in two categories depending on their effect on the original data: (i) *non-perturbative masking* and (ii) *perturbative masking.*

### 2.2.1 Non-perturbative microdata masking

Non-perturbative masking does not rely on distortion of the original data but on partial suppressions or reductions of detail. Some of the methods in this class are usable on both categorical and continuous data, but others are not suitable for continuous data.

Table 2.1 lists the non-perturbative methods described as follows. For each method, the table indicates whether it is suitable for continuous and/or categorical data.

| Methods | Continuous data | Categorical data |
|---|---|---|
| Sampling | | ✓ |
| Global recording | ✓ | ✓ |
| Top and bottom coding | ✓ | ✓ |
| Local suppression | | ✓ |

Table 2.1: Non-preturbative methods vs data types.

**Sampling**

Instead of publishing the original microdata file, what is published is a sample S of the original set of records.

Sampling methods are suitable for categorical microdata, but their adequacy for continuous microdata is less clear in a general disclosure scenario. The reason is that such methods leave a continuous variable $V_i$ unperturbed for all records in $S$. Thus, if variable $V_i$ is present in an external administrative public file, unique matches with the published sample are very likely: indeed, given a continuous variable $V_i$ and two respondents $o_1$ and $o_2$, it is highly unlikely that $V_i$ will take the same value for both $o_1$ and $o_2$ unless $o_1 = o_2$ (this is true even if $V_i$ has been truncated to represent it

digitally).

If, for a continuous identifying variable, the score of a respondent is only approximately known by an attacker (as assumed in [19]), it might make sense to use sampling methods to protect that variable. However, assumptions on restricted attacker resources are perilous and may prove definitely too optimistic if good-quality external administrative files are at hand.

**Global recoding**

For a categorical variable $V_i$, global recoding, a.k.a. generalization, combines several categories to form new (more general) categories, thus resulting in a new $V_i'$ with $|D(V_i')| < |D(V_i)|$, where $|\cdot|$ is the cardinality operator. For a continuous variable, global recoding means replacing $V_i$ by another variable $V_i'$ which is a discretised version of $V_i$. In other words, a potentially infinite range $D(V_i)$ is mapped onto a finite range $D(V_i$"). This is the technique used in the -argus SDL package [20]. This technique is more appropriate for categorical microdata, where it helps disguise records with strange combinations of categorical variables. Global recoding is used heavily by statistical offices.

**Top and bottom coding**

Top and bottom coding is a special case of global recoding which can be used on variables that can be ranked, that is, variables that are continuous or categorical ordinal. The idea is that top values (those above a certain threshold) are lumped together to form a new category. The same is done for bottom values (those below a certain threshold) (see [20] or [21]).

**Local suppression**

If a combination of key variable values (quasi-identifier values) is shared by too few records, it is called an *unsafe combination*, because it may lead to re-identification. Certain values of individual variables are suppressed, that is, replaced with missing

values, with the aim of eliminating unsafe combinations by increasing the set of records agreeing on each combination of key values.

### 2.2.2 Perturbative microdata masking

Perturbative microdata masking methods allow for the release of the entire microdata set, although perturbed values rather than exact values are released. Not all perturbative methods are designed for continuous data; this distinction is addressed further below for each method.

| Methods | Continuous data | Categorical data |
|---|---|---|
| Noise masking | ✓ | |
| Micro-aggregation | ✓ | (✓) |
| Rank swapping | ✓ | (✓) |
| Data shuffling | ✓ | (✓) |
| Rounding | ✓ | |
| Re-sampling | ✓ | |
| PRAM | | ✓ |
| MASSC | | ✓ |

Table 2.2: Perturbative methods vs. data types. ✓- denotes applicable and (✓) - denotes applicable to ordinal categories only.

Most perturbative methods reviewed below (including noise masking (additive and multiplicative), rank swapping, microaggregation and post-randomisation) are special cases of matrix masking. If the original microdata set is $\mathbf{X}$, then the masked microdata set $\mathbf{Z}$ is computed as

$$\mathbf{Z} = \mathbf{AXB} + \mathbf{C},$$

where **A** is a record-transforming mask, **B** is a variable-transforming mask and **C** is a displacing mask or noise [22].

Table 2.2 lists the perturbative methods described as follows. For each method, the table indicates whether it is suitable for continuous and/or categorical data.

**Additive noise masking**

The main noise addition algorithms in the literature are:

- Masking by uncorrelated noise addition.
- Masking by correlated noise addition.
- Masking by noise addition and linear transformation.
- Masking by noise addition and non-linear transformation [23]

As argued in [24], in practice only simple noise addition (two first variants) or noise addition with linear transformation are used. When using linear transformations, a decision has to be made whether to reveal to the data user the parameter $c$ determining the transformations to allow for bias adjustment in the case of subpopulations.

With the exception of the not very practical method of [23], noise addition is not suitable to protect categorical data. On the other hand, it is well suited for continuous data for the following reasons:

(i) It makes no assumptions on the range of possible values for Vi (which may be infinite).

(ii) The noise being added is typically continuous and with mean zero, which suits well continuous original data.

(iii) No exact matching is possible with external files. Depending on the amount of noise added, approximate (interval) matching might be possible.

**Multiplicative noise masking**

One main challenge regarding additive noise with constant variance is that on the one hand small values are strongly perturbed and on the other large values are weakly perturbed. For instance, in a business microdata set the large enterprises - which are much easier to re-identify than the smaller ones - remain still high at risk. A possible way out is given by the multiplicative noise approaches explained as follows.

Let $X$ be the matrix of original numerical data and $W$ the matrix of continuous perturbation variables with expectation 1 and variance $\sigma_W^2 > 0$. The corresponding anonymised data $X^a$ is then obtained by

$$X^a = W \odot X,$$

where $\odot$ is so-called Hadamard product (element-wise matrix multiplication). That is

$$(X^a)_{ij} = \omega_{ij} X_{ij}$$

for each pair $(i, j)$.

**Micro-aggregation**

This method [25] is relevant for continuous variables, such as in business survey microdata, and in its basic form consists of ordering the values of each variable and forming groups of a specified size $k$ (the first group contains the $k$ smallest values, the second group the next $k$ smallest values and so on). The method replaces the values by their group means, separately for each variable. An advantage of the method is that the modification to the data will usually be greatest for outlying values, which might also be deemed the most risky. It is difficult, however, for the user to assess the biasing impact of the method on analyses.

For overview of others SDL methods and their details see [1].

# CHAPTER 3

# SDL METHODS OVERVIEW: TABULAR DATA

As we mentioned before, the tabular data is one of the main type of data that is needed to be protected and that's SDC is dealing with. There are two types of tabular data: (i) magnitude tabular data, and (ii) frequency tables. Let provide the basic SDC methods corresponding to those types of data.

## 3.1 Disclosure risk

The statistical agencies usually define the disclosure risk for tabular data through two assessment: (i) primary sensitive cells and (ii) secondary risk assessment.

**Disclosure risk assessment I: Primary sensitive cells**

Considering the different type of intruder scenarios, statistical agencies have developed some safety rules as measures to assess disclosure risks. In table 3.1 we provide the main definition for sensitivity rule.

**Disclosure risk assessment II: Secondary risk assessment**

When some small, primary confidential cells within detailed tables have to be protected, it may still be possible to publish sums for larger groups, i.e., the margins of those detailed tables. However, we must make sure that protection of the primary sensitive cells cannot be undone, by some differencing.

Modelling secondary disclosure risks requires that apart from its inner cells margins and overall totals are considered to be part of a table. From this assumption, it follows that there is always a linear relationship between cells of a table. Other cells (so-called complementary or secondary cells) must be suppressed, or be otherwise manipulated in order to prevent the value of the protected sensitive cell being recalculated through, e.g. differencing.

When a table is protected by cell suppression, it is always possible for any par-

| Rule: | Definition: |
|---|---|
| | A cell is considered *unsafe*, when... |
| Minimum frequency rule | the number of contributors to the cell is less than a pre-specified minimum frequency $n$ (the common choice is $n = 3$). |
| $(n, k)$-dominance rule | the sum of the $n$ largest contributions exceeds $k\%$ of the cell total, i.e., $x_1 + \ldots + x_n > k/100X$. |
| $p\%$ rule | the cell total minus the two largest contributions $x_1$ and $x_2$ is less than $p\%$ of the largest contribution, i.e., $Xx_n \ldots x_2 x_1 < p/100x_1$. |

Table 3.1: Sensitivity rules.

ticular suppressed cell of a table to derive upper and lower bounds for its true value, by making use of the linear relations between published and suppressed cell values. This holds for tables with non-negative values. In case of tables containing negative values as well, it holds when some (possibly tight) lower bound other than zero is available to data users in advance of publication for all cells in the table. The interval given by these bounds is called the *feasibility interval*.

A general mathematical statement for the linear programming problem to compute upper and lower bounds for the suppressed entries of a table is given in [26]. Within a disclosure control process, computing the bounds of the feasibility intervals subject to a set of table relations is referred to as an *audit* of a protected table.

Note that feasibility intervals can in principle also be computed using this ap-

proach, when cell perturbation (as opposed to cell suppression) is used to protect the table. To be able to do this, the user should know something about the perturbation method. In general, the user should be given information such that she can deduce meaningful (i.e. rather narrow) a priory intervals of each perturbed cell. Those intervals are referred to as *implicit intervals* in the following. The table relations can then be used to derive tighter a posteriori bounds of the perturbed cells.

The protection provided by a set of suppressions (the suppression pattern), or by a perturbation technique that supplies users with implicit intervals, should only be considered valid if the bounds for the feasibility interval of any sensitive cell cannot be used to deduce bounds on an individual respondents contribution to that cell that are too close according to the sensitivity rule employed. For a mathematical statement of that condition, we determine safety bounds for primary suppressions. We call the deviation between those safety bounds and the true cell value *upper and lower protection levels*. The formulas of Table 3.2 can be used to compute upper protection levels in case of concentration rules. Out of symmetry considerations, the lower protection level is often set identical to the upper protection level. The bounds of the protection interval are computed by subtracting the lower protection from and adding the upper protection level to the cell value. the minimum frequency rules should only be used instead of concentration rules if it is enough to prevent exact disclosure only. In such a case, the protection levels should also be chosen such that exact disclosure is prevented. That is, a minimal protection level is sufficient.

If the distance between the upper bound of the feasibility interval and the true value of a sensitive cell is below the upper protection level computed according to the formulas of Table 3.2, then this upper bound could be used to derive an estimate for individual contributions of the sensitive cell that is too close according to the safety rule employed, which can easily be proven along the lines of [27]. We say

| Sensitivity Rule | Upper protection level |
|---|---|
| $(1, k)$ rule | $(100/k)x_1 - X$ |
| $(n, k)$ rule | $(100/k)(x_1 + x_2 + \ldots + x_n) - X$ |
| $p\%$ rule | $(p/100)x_1 - (X - x_1 - x_2)$ |
| $(p, q)$ rule | $r(p/q)x_1 - (X - x_1 - x_2)$ |

Table 3.2: Upper protection levels.

then that this cell is subject to *upper inferential disclosure*. More generally, if the feasibility interval for a cell does not cover its protection interval, we say that the cell is *underprotected*, not *sufficiently protected* or *subject to inferential disclosure*. If the feasibility interval consists of only the true value of the sensitive cell, we say that the cell is subject to *exact disclosure*.

The distance between the upper bound of the feasibility interval and the true value of a sensitive cell must exceed the upper protection level; otherwise the sensitive cell is not properly protected. This safety criterion is a necessary, but not always sufficient criterion for proper protection. It is a sufficient criterion when the largest respondent makes the same contribution also within the combination of suppressed cells within the same aggregation (a row or column relation of the table, for instance), and when no individual contribution of any respondent (or coalition of respondents) to such a combination of suppressed cells is larger than the second largest respondents (or coalitions) contribution.

## 3.2 SDL methods for Magnitude tabular data

Statistical magnitude tables display sums of observations of a quantitative variable where each sum relates to a group of observations defined by categorical variables observed for a set of respondents.

Respondents are typically companies but can also be individuals or households, etc. The categorical *grouping variables* typically give information on geography or economic activity or size, etc. of the respondents. The *cells* of a table are defined by cross-combinations of the grouping variables.

Each *table cell* presents a sum of a quantitative *response* variable such as income, turnover, expenditure, sales, number of employees, number of animals owned by farms, etc. These sums are the *cell values* of a magnitude table. The individual observations of the variable (for each individual respondent) are the *contributions* to the cell value.

The *dimension* of a table is given by the number of grouping variables used to specify the table.We say that a table contains *margins* or *marginal cells*, if not all cells of a table are specified by the same number of grouping variables. The smaller the number of grouping variables, the higher the *level* of a marginal cell.

Basically, there are two different classes of protection methods for tabular data: (1) pre-tabular and (2) post-tabular methods. Pre-tabular methods manipulate the microdata before they are summed up for tabulation and hence do not depend on any particular tabulation. Post-tabular methods are applied after tabulation.

Basically, any of the microdata protection methods of Chapter 2 could be considered as a pre-tabular protection method for tables.

As pointed out in [36], information loss concepts for tabular data are different from those for microdata. This is due to the fact that tables are often seen as final products, and not so much as a starting point for further analysis. This conception is probably even more prominent in case of magnitude tables.

In order to find a good balance between protection of individual response data and provision of information, it is necessary to somehow rate the information loss connected to a cell that is suppressed or perturbed. By doing so, we can try to control

the protection process in order to achieve optimal behaviour of the algorithms. The main SDC method that is used for magnitude tabular data are presented in Table 3.3.

| Type of method | Perturbative | Non-perturbative |
|---|---|---|
| Pre-tabular | Multiplicative noise | Global recoding |
| Post-tabular | Controlled tabular adjustment | Cell suppression |

Table 3.3: Classification of protection methods for tabular data.

**Global recoding**

In the context of tabular data protection, global recoding means that several categories of a categorical variable are collapsed into a single one. This will reduce the amount of detail of any table using this variable as spanning variable. Usually, as a result the number of unsafe cells in the tables will be reduced.

Using only global recoding to protect all sensitive cells in a table often results in huge loss of data utility. While it is certainly not an uncommon strategy in official statistics, it is hence usually applied in combination with other protection methods like cell suppression. Another reason to combine this method with other protection methods is that often publication requirements on the table design do not allow for global recoding. Moreover, it is usually in contrast with the aim of many statistical agencies to squeeze out the maximum amount of information of the data in the tables they release.

**Cell suppression**

For business statistics, the most popular method for tabular data protection is *cell suppression*. In tables protected by cell suppression, all values of cells for which a disclosure risk has been established are eliminated from the publication. In practice, this means that those values will be replaced by a some special symbol.

A cell suppression procedure involves two steps. Firstly, in the primary riskassessment step, all primary sensitive cells are determined. In the second step, some non-sensitive cells are selected as secondary suppressions to protect the primary sensitive cells from disclosure by differencing. The joint set of primary and secondary suppressions is called a *suppression pattern*. Cell suppression is a protection technique that requires a very careful evaluation of secondary disclosure risks.For a mathematical statement of the secondary cell suppression problem, see, e.g.,[26].

**Multiplicative noise**

[29] proposed pre-tabular multiplicative noise for the protection of enterprise tabular data. [30] notes that such a methodology is used at the US Census Bureau for tabular magnitude data protection for several data products. A research report [31] examines statistical properties of two variants. According to the simple variant, multiplicative noise with a mean of one and constant variance is assigned to the microdata. In this case, the conditional (unit level) coefficient of variance of the noisy (micro)data is a constant, defined by the noise variance. For the second variant, a balanced noise method (see [32]), the research report proves that for any set of units, the perturbed total is an unbiased estimate of the original total. Moreover, it states that the balancing mechanism works indeed, i.e., it reduces the noise variance of the cell totals in a reference table.

**Controlled tabular adjustment**

*Controlled tabular adjustment* or CTA is a relatively new protection method for magnitude tabular data, suggested for instance in [33], [34], and [35].

CTA methodology aims at finding the closest additive table to the original table ensuring that adjusted values of all confidential cells are safely away from their original value (considering the protection intervals) and that the adjusted values are within a certain range of the real values.

Thus the learning of CTA for tabular data was the goal of this thesis the more detailed description of this method will be presented in Chapter 5.

## 3.3 SDL methods for Frequency tables

Traditionally, frequency tables have been the main method of dissemination for census and social data by National Statistical Institutes (NSIs). These tables contain counts of people or households with certain social characteristics. Frequency tables are also used for business data where characteristics are counted, such as the number of businesses. Because of their longer history, there has been relatively more research on protecting frequency tables, as compared with newer output methods such as microdata.

There are a variety of disclosure control methods which can be applied to tabular data to provide confidentiality protection. The choice of which method to use needs to balance how the data are used, the operational feasibility of the method and the disclosure control protection it offers. SDC methods can be divided into three categories which will be discussed in turn further: (1) those that adjust the data before tables are designed (pre-tabular), (2) those that determine the design of the table (table redesign) and (3) those that modify the values in the table (post-tabular). Further information on SDC methods for frequency tables can also be found in [36] and [37].

**Pre-tabular methods**

Pre-tabular disclosure control methods are applied to microdata before they are aggregated and output in frequency tables. These methods include: record swapping, overimputation, data switching, PRAM and sampling. A key advantage of pre-tabular methods is that the output tables are consistent and additive since all outputs are created from protected microdata. Pretabular methods by definition only need to be

applied once to the microdata and after they are implemented for a microdata set (often in conjunction with threshold or sparsity rules) they can be used to allow flexible table generation. This is because pre-tabular methods provide some protection against disclosure by differencing and any uncovered slivers will have already had SDC protection applied.

Disadvantages of pre-tabular techniques are that one must have access to the original microdata. Also, a high level of perturbation may be required in order to disguise all unsafe cells. Pre-tabular methods have the potential to distort distributions in the data, but the actual impact of this will depend on which method is used and how it is applied. It may be possible to target pre-tabular methods towards particular areas or sensitive variables. Generally pre-tabular methods are not as transparent to users of the frequency tables and there is no clear guidance that can be given in order to make adjustments in their statistical analysis for this type of perturbation.

**Table re-design methods**

Table redesign is recommended as a simple method that can minimize the number of unsafe cells in a table and preserve original counts. It can be applied alongside post-tabular or pre-tabular disclosure control methods, as well as being applied on its own. As an additional method of protection, it has been used in many NSIs including the UK and New Zealand. As table redesign alone provides relatively less disclosure control protection than other methods, it is often used to protect sample data, which already contains some protection from the sampling process.

The advantages of table redesign methods are that original counts in the data are not damaged and the tables are additive with consistent totals. In addition, the method is simple to implement and easy to explain to users. However, the detail in the table will be greatly reduced, and if many tables do not pass the release criteria it may lead to user discontent.

### Post-tabular methods

Statistical disclosure control methods that modify cell values within tabular outputs are referred to as post-tabular methods. Such methods are generally clear and transparent to users, and are easier to understand and account for in analyses, than pre-tabular methods. However, post-tabular methods suffer the problem that each table must be individually protected, and it is necessary to ensure that the new protected table cannot be compared against any other existing outputs in such a way which may undo the protection that has been applied. In addition, post-tabular methods can be cumbersome to apply to large tables. The main post-tabular methods include cell suppression, cell perturbation and rounding.

# CHAPTER 4

# OVERVIEW OF CONES AND INTERIOR POINT METHOD

In this thesis we are going to present the results of controlled tabular adjustment (CTA) as SDL method for tabular data. Since formulation of CTA leads to solve optimization problem such as Liner Programing, Quadratic Programing or Conic Programing Problems, we first give a brief overview of these optimization problems and models.

## 4.1 Linear Programing

### 4.1.1 Interior Point Method for Linear Programing

Linear Programing (LP) problem in the standard form can be formulated as: Given the data, vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and matrix $A \in \mathbb{R}^{m\times n}$, find a vector $x \in \mathbb{R}^n$ that solves the problem:

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & Ax = b, \\ & x \geq 0. \end{aligned} \tag{4.1}$$

The vector $x \in \mathbb{R}^n$ is called a vector of primal variables and the set $F_p = \{x : \ Ax = b, \ x \geq 0\}$ is called a primal feasible region.

The corresponding dual problem is then given by:

$$\begin{aligned} \max \quad & b^T y \\ s.t. \quad & A^T y + s = c, \\ & s \geq 0. \end{aligned} \tag{4.2}$$

The vector $y \in \mathbb{R}^m$ is called a vector of dual variables and the vector $s \in \mathbb{R}^n$ is called a vector of dual slack variables. The set $F_d = \left\{(y, s) : \ A^T y + s = c, \ s \geq 0\right\}$ is called a dual feasible region.

For years the basic method for solving the Linear Programing (LP) problems was the Dantzigs Simplex Method [38], but nowadays an effective alternative is the Interior Point Method as a basic solver for LP problems. Let introduce the main concepts of this approach.

The Karuch-Kuhn-Tucker (KKT) conditions for LP are:

$$\begin{aligned} A^T y + s - c = 0,\ s \geq 0, \quad &\longleftarrow \text{Dual feasibility} \\ b - Ax = 0,\ x \geq 0, \quad &\longleftarrow \text{Primal feasibility} \\ Xs = 0. \quad &\longleftarrow \text{Complementarity} \end{aligned} \tag{4.3}$$

The standard method of choice for finding an approximate solution of the system (4.3) is the Modified (damped) Newtons Method (MNM), that is, the Newtons Method with line search. However in order for MNM to work it is necessary to perturb Complementarity condition as $Xs = \mu e$, where $\mu = \frac{x^T s}{n}$ and $e \in \mathbb{R}^n$ is a vector of ones. One way to justify this perturbation is based on theory of Barrier Methods [43]

The system (4.3) can be viewed as the system parameterized in $\mu > 0$. This parameterized system has a unique solution for each $\mu > 0$ if $\text{rank}(A) = m$. The set of $\mu$ -centers gives a homotopy path, which is called the central path of (4.1) and (4.2) respectively. The limiting property ($\mu \to 0$) of the central path mentioned by [39], [40],[41] leads naturally to the main idea of the iterative methods for solving (4.1) and (4.2): trace the central path while reducing $\mu$ at each iteration.

It is known that NM may not converge in general. The main achievement of IPMs is that NM is used in a such way that guarantees global convergence.

The choice of a step size $\alpha_k$ of MNM is the key to proving good global convergence of the method. The statement that approximate solutions of (4.3), or iterates should not be "too far" from the central path is formalized by introducing the horn neighborhood of the central path. The horn neighborhoods of the central path can

be defined using different norms

$$N_2(\beta) = \{(x,s) : \ \|Xs - \mu e\|_2 \leq \beta\mu\}, \tag{4.4}$$

$$N_\infty(\beta) = \{(x,s) : \ \|Xs - \mu e\|_\infty \leq \beta\mu\}, \tag{4.5}$$

The infeasible IPM for LP can now be summarized as follows.

**Algorithm (IPM)**

*Initialization*

1. Choose $\beta, \gamma \in (0,1)$ and $(\varepsilon_P, \varepsilon_D, \varepsilon_G) > 0$. Choose $(x^0, y^0, s^0)$ such that $(x^0, s^0) > 0$ and $\|X^0 s^0 - \mu_0 e\| \leq \beta\mu_0$, where $\mu_0 = \frac{(x^0)^T s^0}{n}$.

2. Set $k = 0$.

   *Step*

3. Set $r_P^k = b - Ax^k, \ r_D^k = c - A^T y^k - s^k, \ \mu_k = \frac{(x^k)^T s^k}{n}$.

4. Check termination. If $\|r_P^k\| \leq \varepsilon_P, \ \|r_D^k\| \leq \varepsilon_D, \ (x^k)^T s^k \leq \varepsilon_G$, then terminate.

5. Compute the direction by solving the system

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} d_x \\ d_y \\ d_s \end{bmatrix} = \begin{bmatrix} r_P^k \\ r_D^k \\ -X^k s^k + \gamma\mu_k e \end{bmatrix}$$

   .

6. Compute the step size

$$\alpha_k = \max\{\alpha' : \|X(\alpha)s(\alpha) - \mu(\alpha)e\| \leq \beta\mu(\alpha), \ \forall\alpha \in [0, \alpha']\}$$

   , where

$$x(\alpha) = x^k + \alpha d_x, \ s(\alpha) = s^k + \alpha d_s, \ \mu(\alpha) = \frac{x^T(\alpha)s(\alpha)}{n}.$$

   .

7. Update $x^{k+1} = x^k + \alpha_k d_x,\ \ y^{k+1} = y^k + \alpha_k d_y,\ \ s^{k+1} = s^k + \alpha_k d_s$.

8. Set $k = k + 1$ and go to step 3.

More details and a simplified version of this IPM is described in [42]. Important fact is that the system in step 5 can be significantly reduced. This reduction is known as 'normal equations'.

### 4.1.2 Extension to Quadratic Programing

The algorithm of the previous subsection can be extend to the convex quadratic programing (QP) problems.

Consider a QP problem in the standard form:

$$\begin{aligned} \min \quad & \frac{1}{2} x^T Q x + c^T x \\ s.t. \quad & Ax \geq b, \\ & x \geq 0, \end{aligned} \tag{4.6}$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetric positive semidefinite matrix, vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $x \in \mathbb{R}^n$ and matrix $A \in \mathbb{R}^{m \times n}$.

Then the KKT condition for (4.6) are:

$$\begin{aligned} 0 \leq x \ \ \perp \ \ & Qx - A^T u + c \geq 0 \\ 0 \leq u \ \ \perp \ & Ax - b \geq 0. \end{aligned} \tag{4.7}$$

By introducing slack variables $v$ and $s$ and by defining diagonal matrices $X, S, U, V$ from components of $x, s, u, v$ we can rewrite the KKT condition in a form similar to

(4.3):

$$\begin{aligned} Qx - A^T u - s &= -c, \\ Ax - v &= b, \\ XSe &= 0, \\ UVe &= 0, \\ x \geq 0,\ s \geq 0,\ u \geq 0,\ v &\geq 0. \end{aligned} \tag{4.8}$$

As for LP algorithm providing above, we design IPM by applying MNM to the linear system (4.8). The difference between LP and QP cases is how we define perturbation $\mu = \frac{1}{m+n}(x^T s + u^T v)$ and the system in Step 5 of IPM's Algorithm is defined as:

$$\begin{bmatrix} Q & -I & -A^T & 0 \\ A & 0 & 0 & -I \\ S^k & X^k & 0 & 0 \\ 0 & 0 & V^k & U^k \end{bmatrix} \begin{bmatrix} d_x \\ d_s \\ d_u \\ d_v \end{bmatrix} = - \begin{bmatrix} Qx^k - A^T k - s^k + c \\ Ax^k - v^k - b \\ X^k S^k e - \gamma_k \mu_k e \\ U^k V^k e - \gamma_k \mu_k e \end{bmatrix}.$$

Similarly as in LP case this system can be reduced.

## 4.2 Overview of conic optimization

In both LP and QP the variables are required to be non-negative, that is they belong to the non-negative orthant. Non-negative orthant is the basic example of symmetric cone. Thus, LP and QP are examples of a more general optimization problem called Conic Optimization (CO) Problem.

The conic optimization model can be written in standard form as

$$\min_x c^T x :\ Ax = b,\ x \in \mathcal{K}, \tag{4.9}$$

where $\mathcal{K}$ is a symmetric cone. Conic optimization problems can be solved using appropriate forms of IPM.

In sequel we give a definition of a symmetric cone and list several examples of symmetric cones that appear frequently in CO.

Lets mention some basic definition of cones.

**Definition 4.1** (Convex set)**.** *A set $\mathcal{K}$ is convex if for any $x, y \in \mathcal{K}$ and any $\alpha$ with $0 \leq \alpha \leq 1$, we have $\alpha x + (1-\alpha)y \in \mathcal{K}$.*

**Definition 4.2** (Cone)**.** *A set $\mathcal{K}$ is called a cone if for every $x \in \mathcal{K}$ and $\alpha \geq 0$, we have $\alpha x \in \mathcal{K}$.*

Therefore, a set $\mathcal{K}$ is called a *convex cone* if it is convex and a cone, which means that for any $x, y \in \mathcal{K}$ and $\alpha, \beta \geq 0$, we have $\alpha x + \beta y \in \mathcal{K}$.

**Definition 4.3** (Dual cone)**.** *Let $\mathcal{K}$ be a cone. The set*

$$\mathcal{K}^* := \{y : \langle x, y\rangle \geq 0, \text{ for all } \ x \in \mathcal{K}\}$$

*is called a dual cone of $\mathcal{K}$.*

As the name suggests, $\mathcal{K}^*$ is a cone, and it is always convex, even when the original cone is not. If cone $\mathcal{K}$ and its dual $\mathcal{K}^*$ coincide, we say that $\mathcal{K}$ is self-dual. In particular, this implies that $\mathcal{K}$ has a nonempty interior and does not contain any straight line (i.e., it is pointed).

**Definition 4.4** (Homogeneity)**.** *The convex cone $\mathcal{K}$ is said to be homogeneous if for every pair $x, y \in int\mathcal{K}$, there exists an invertible linear operator $g$ for which $g\mathcal{K} = \mathcal{K}$ and $gx = y$.*

In fact, the above linear operator $g$ is an automorphism of the cone $\mathcal{K}$.

**Definition 4.5.** *The convex cone $\mathcal{K}$ is said to be symmetric if it is self-dual and homogeneous.*

Below, we list the important examples of symmetric cones:

1. The linear cone or non-negative orthant:

$$\mathcal{K} = \mathbb{R}^n_+ := \{x \in \mathbb{R}^n : x_i \geq 0,\ i = 1, \ldots, n\}.$$

   CO with this cone leads to the standard LP problem.

2. The positive semidefinite cone:

$$\mathcal{K} = \mathcal{S}^n_+ := \left\{X \in \mathcal{S}^n_+ :\ X \succeq 0\right\},$$

   where $\succeq$ means that $X$ is positive semidefinite matrix and $\mathcal{S}^{\backslash}$ is a set of symmetric $n$-dimensional matrices.

   CO with this cone leads to the semidefinite programing problem.

3. The quadratic or second-order cone:

$$\mathcal{K} = \mathcal{Q}^n := \left\{(x, t) \in \mathbb{R}^n_+ :\ t \geq \|x\|_2\right\}.$$

   CO with this cone leads to the second order cone programing problem.

We can also provide the definition of the symmetric cone in terms of Euclidean Jordan Algebra.

**Definition 4.6** (Bilinear map)**.** *Let $\mathcal{J}$ be a finite-dimensional vector space over $\mathbb{R}$. A map $\circ : \mathcal{J} \times \mathcal{J} \mapsto \mathcal{J}$ is called bilinear if for all $x, y, z \in \mathcal{J}$ and $\alpha, \beta \in \mathbb{R}$:*

*(i)* $(\alpha x + \beta y) \circ z = \alpha(x \circ z) + \beta(y \circ z)$;

*(ii)* $x \circ (\alpha y + \beta z) = \alpha(x \circ y) + \beta(x \circ z)$.

**Definition 4.7** ($\mathbb{R}$-algebra)**.** *A finite-dimensional vector space $\mathcal{J}$ over $\mathbb{R}$ is called an algebra over $\mathbb{R}$ if a bilinear map from $\mathcal{J} \times \mathcal{J}$ into $\mathcal{J}$ is defined.*

**Definition 4.8** (Jordan algebra)**.** *Let $\mathcal{J}$ be a finite-dimensional $\mathbb{R}$-algebra along with a bilinear map $\circ : \mathcal{J} \times \mathcal{J} \mapsto \mathcal{J}$. Then $(\mathcal{J}, \circ)$ is called a Jordan algebra if for all $x, y \in \mathcal{J}$ the following holds:*

*(i)* $x \circ y = y \circ x$ *(Commutativity);*

*(ii)* $x \circ (x^2 \circ y) = x^2 \circ (x \circ y)$*, where* $x^2 = x \circ x$ *(Jordan's Axiom).*

Here we assume that $(\mathcal{J}, \circ)$ is a Jordan algebra, which we simply denote as $\mathcal{J}$.

**Definition 4.9** (Euclidean Jordan algebra)**.** *We consider a finite-dimensional Jordan algebra $\mathcal{J}$ over $\mathbb{R}$ and assume the existence of the identity element $e$. The Jordan algebra $\mathcal{J}$ is said to be Euclidean if there exists a positive definite symmetric bilinear form on $\mathcal{J}$ which is associative; in other words, there exists an inner product denoted by $\langle \cdot, \cdot \rangle$, such that $\langle x \circ y, z \rangle = \langle x, y \circ z \rangle$, for all $x, y, z \in \mathcal{J}$.*

**Definition 4.10** (Symmetric cone)**.** *Cone is symmetric if it is a cone of squares of a certain Euclidean Jordan algebra $\mathcal{J}$:*

$$\{x^2 = x \circ x | x \in \mathcal{J}\}$$

.

It can be shown that definitions 4.5 and 4.10 are equivalent. The definitions of Euclidean Jordan algebra and symmetric cones, their properties and their applications in conic optimization can be found in [44].

# CHAPTER 5

# CONTROLLED TABULAR ADJUSTMENT MODEL FOR SDL

*Minimum-distance controlled tabular adjustment* (CTA) methodology was first introduced in [49, 52]. As indicated in [48], CTA can be formulated as the following problem: Given a table with sensitive cells, compute the closest safe table in which sensitive cells are modified to avoid re-computation, and the remaining cells are minimally adjusted to satisfy the table equations. The closeness of the original and modified table is measured by the weighted distance between the tables with respect to a certain norm. Most commonly used norms are $\ell_1$ and $\ell_2$ norms. Thus, the problem can be formulated as a minimization problem with the objective function being a particular weighted distance function and constraints being table equations and lower and upper bounds on the cell values.

In general, CTA is Mixed Integer Optimization Problem (MIOP) which is a difficult problem to solve especially for the large dimension problems. A priori fixing the values of binary variables reduces the problem to the continuous optimization problem which is easier to solve, however, the quality of the solution may be reduced. In addition, the values of the binary variables have to be assigned carefully otherwise the problem may become infeasible [50, 51].

## 5.1 Formulation of the General CTA Model

The following CTA formulation is given in [48]: Given the following set of parameters:

(i) A set of cells $a_i, i \in \mathcal{N} = \{1, \ldots, n\}$. The vector $a = (a_1, \ldots, a_n)^T$ satisfies certain linear system $Aa = b$ where $A \in \mathbb{R}^{m \times n}$ is an $m \times n$ matrix and and $b \in \mathbb{R}^m$ is $m$-vector.

(ii) A lower, and upper bound for each cell, $l_{a_i} \leq a_i \leq u_{a_i}$ for $i \in \mathcal{N}$, which are

considered known by any attacker.

(iii) A set of indices of sensitive cells, $\mathcal{S} = \{i_1, i_2, \ldots, i_s\} \subseteq \mathcal{N}$.

(iv) A lower and upper protection level for each sensitive cell $i \in \mathcal{S}$ respectively, $lpl_i$ and $upl_i$, such that the released values must be outside of the interval $(a_i - lpl_i,\ a_i + upl_i)$.

(v) A set of weights, $w_i, i \in \mathcal{N}$ used in measuring the deviation of the released data values from the original data values.

A CTA problem is a problem of finding values $z_i,\ i \in \mathcal{N}$, to be released, such that $z_i,\ i \in \mathcal{S}$ are safe values and the weighted distance between released values $z_i$ and original values $a_i$, denoted as $\|z - a\|_{l(w)}$, is minimized, which leads to solving the following optimization problem

$$\begin{aligned} \min_{z} \quad & \|z - a\|_{l(w)} \\ s.t. \quad & Az = b, \\ & l_{a_i} \leq z_i \leq u_{a_i},\ i \in \mathcal{N}, \\ & z_i,\ i \in \mathcal{S} \text{ are safe values.} \end{aligned} \tag{5.1}$$

As indicated in the assumption (iv) above, safe values are the values that satisfy

$$z_i \leq a_i - lpl_i \text{ or } z_i \geq a_i + upl_i,\ i \in \mathcal{S}. \tag{5.2}$$

By introducing a vector of binary variables $y \in \{0,1\}^s$ the constraint (5.2) can be written as

$$\begin{aligned} z_i \geq \quad & -M\,(1 - y_i) + (a_i + upl_i)\,y_i, \quad i \in \mathcal{S}, \\ z_i \leq \quad & My_i + (a_i - lpl_i)\,(1 - y_i)\,, \qquad i \in \mathcal{S}, \end{aligned} \tag{5.3}$$

where $M \gg 0$ is a large positive number. Constraints (5.3) enforce the upper safe value if $y_i = 1$ or the lower safe value if $y_i = 0$.

Replacing the last constraint in the CTA model (5.1) with (5.3) leads to a mixed integer convex optimization problem (MIOP) which is in general a difficult problem to solve; however, it provides solutions with high data utility. The alternative approach is to fix binary variables up front which leads to a CTA that is acontinuous convex optimization problem. The continuous CTA may be easier to solve; however, the obtained solution may have a lower data utility. Furthermore, a wrong assignment of binary variables may result in the problem being infeasible. Strategies on how to avoid this difficulty are discussed in [50, 51].

In this thesis we consider a continuous CTA where binary variables are fixed and vector $z$ is replaced by the vector of *cell deviations*

$$x = z - a. \tag{5.4}$$

The CTA (5.1) with constraints (5.3) reduces to the following convex optimization problem:

$$\begin{aligned} \min_{x} \quad & \|x\|_{l(w)} \\ s.t. \quad & Ax = 0, \\ & l \leq x \leq u \ , \end{aligned} \tag{5.5}$$

where upper and lover bounds for $x_i,\ i \in \mathcal{N}$ are defined as follows:

$$l_i = \begin{cases} upl_i & \text{if } i \in \mathcal{S} \text{ and } y_i = 1 \\ l_{a_i} - a_i & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 0) \end{cases} \tag{5.6}$$

$$u_i = \begin{cases} -lpl_i & \text{if } i \in \mathcal{S} \text{ and } y_i = 0 \\ u_{a_i} - a_i & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 1)\,. \end{cases} \tag{5.7}$$

The two most commonly used norms in problem (5.5) are the $\ell_1$ and $\ell_2$ norms. For the $\ell_2$-norm the problem, (5.5) reduces to the following $\ell_2$-CTA model which is a

QP problem:

$$\begin{aligned} \min_{x} \quad & \sum_{i=1}^{n} w_i x_i^2 \\ s.t. \quad & Ax = 0, \\ & l \leq x \leq u \ . \end{aligned} \tag{5.8}$$

For the $\ell_1$-norm the problem, (5.5) reduces to the following $\ell_1$-CTA model:

$$\begin{aligned} \min_{x} \quad & \sum_{i=1}^{n} w_i \left| x_i \right| \\ s.t. \quad & Ax = 0, \\ & l \leq x \leq u \ . \end{aligned} \tag{5.9}$$

The above $\ell_1$-CTA model (5.9) is a convex optimization problem; however, the objective function is not differentiable at $x = 0$. Since most of the algorithms, including IPMs, require differentiability of the objective function, problem (5.9) needs to be reformulated. The reformulations that have been considered in [48] are reviewed in the next section.

## 5.2 LP and Pseudo-Huber Formulation of $\ell_1$-CTA

The $\ell_2$-CTA model (5.8) is a standard QP problem that can be efficiently solved using IPM or other methods. However, as noted at the end of the previous section, the $\ell_1$-CTA model (5.9) needs reformulation in order to be efficiently solved by IPM or some other method. The standard reformulation is the transformation of model (5.9) to the following LP model:

$$\begin{aligned} \min_{x^-,x^+} \quad & \sum_{i=1}^{n} w_i \left(x_i^+ + x_i^-\right) \\ s.t. \quad & A\left(x_i^+ - x_i^-\right) = 0, \\ & l^+ \le x^+ \le u^+, \\ & l^- \le x^- \le u^-, \end{aligned} \tag{5.10}$$

where

$$x^+ = \begin{cases} x & \text{if } x \ge 0 \\ 0 & \text{if } x < 0, \end{cases} \qquad x^- = \begin{cases} 0 & \text{if } x > 0 \\ -x & \text{if } x \le 0, \end{cases} \tag{5.11}$$

and lower and upper bounds for $x_i^-$ and $x_i^+,\ i \in \mathcal{N}$ are as follows:

$$\begin{aligned} l_i^+ &= \begin{cases} upl_i & \text{if } i \in \mathcal{S} \text{ and } y_i = 1 \\ 0 & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 0) \end{cases} \\ u_i^+ &= \begin{cases} 0 & \text{if } i \in \mathcal{S} \text{ and } y_i = 0 \\ u_{a_i} - a_i & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 1) \end{cases} \\ l_i^- &= \begin{cases} lpl_i & \text{if } i \in \mathcal{S} \text{ and } y_i = 0 \\ 0 & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 1) \end{cases} \\ u_i^- &= \begin{cases} 0 & \text{if } i \in \mathcal{S} \text{ and } y_i = 1 \\ a_i - l_{a_i} & \text{if } (i \in \mathcal{N} \setminus \mathcal{S}) \text{ or } (i \in \mathcal{S} \text{ and } y_i = 0). \end{cases} \end{aligned} \tag{5.12}$$

Problem $\ell_1$-CTA (5.10) is an LP problem; however, it has twice the number of variables as the QP problem (5.8) and twice the number of box constraints. As indicated in [48], the splitting of the variables $x = x^+ - x^-$ and the increased dimension of the model may cause problems. In order to overcome these difficulties in [48] it was suggested to use a regularization of problem (5.9) by approximating absolute value

with the Pseudo-Huber function that has the same number of variables as in the QP formulation (5.8).

The original Huber function $\varphi_\delta : \mathbb{R} \longrightarrow \mathbb{R}_+$ is defined as

$$\varphi_\delta(x_i) = \begin{cases} \frac{x_i^2}{2\delta} & |x_i| \leq \delta \\ |x_i| - \frac{\delta}{2} & |x_i| \geq \delta. \end{cases} \tag{5.13}$$

It approximates $|x_i|$ for small values of $\delta > 0$; the smaller the $\delta$, the better the approximation.The Huber function is continuously differentiable; however, the second derivative is not continuous at $|x_i| = \delta$ which may cause problems when this function is used in second order optimization algorithms, such as IPMs. Hence, it is better to consider the Pseudo-Huber function $\phi_\delta : \mathbb{R} \longrightarrow \mathbb{R}_+$

$$\phi_\delta(x_i) = \sqrt{\delta^2 + x_i^2} - \delta \tag{5.14}$$

whose first and second derivatives are bounded and Lipschitz continuous [**?**]. Again, the smaller the $\delta$ the better the approximation.

Now, the $\ell_1$-CTA problem (5.9) can be approximated by the following convex optimization problem

$$\begin{aligned} \min_x \quad & \textstyle\sum_{i=1}^n w_i \phi_\delta(x_i) \\ s.t. \quad & Ax = 0, \\ & l \leq x \leq u \ . \end{aligned} \tag{5.15}$$

The advantage of the Pseudo-Huber-CTA model (5.15) is that it has the same number of variables as $\ell_2$ - CTA and the same feasible region, the only difference is that the quadratic objective function is replaced by a strictly convex function.

Optimization problems (5.8), (5.10) and (5.15) can be solved with appropriate versions of the Interior-Point Methods (IPM), that was described in previous chapter.

## 5.3 SOC Formulation of Pseudo-Huber and $\ell_1$ CTA

In this section we investigate how Pseudo-Huber and $\ell_1$ CTA can be formulated as SOC models.

We define CO problem as (4.9). In what follows, we present a reformulation of Pseudo-Huber-CTA problem (5.15) as a SOC problem. Consider Pseudo-Huber Function (5.14)

$$\phi_\delta(x_i) = \sqrt{\delta^2 + x_i^2} - \delta.$$

Let's define

$$t_i := \sqrt{\delta^2 + x_i^2} \quad \text{and} \quad y_i := \delta, \quad i = 1, \ldots, n \ . \tag{5.16}$$

Hence, we have

$$t_i = \sqrt{x_i^2 + y_i^2}$$

which is the boundary of the second-order (quadratic) cone

$$\mathcal{K}_i = \left\{ (x_i, y_i, t_i) \ \in \mathbb{R}^3 \ : \ t_i \geq \sqrt{x_i^2 + y_i^2} \right\} \ .$$

Now, the reformulation of the Pseudo-Huber-CTA (5.15) as a SOC problem follows

$$\begin{array}{rl} \min\limits_{x} & \sum_{i=1}^{n} w_i \left(t_i - y_i\right) \\ s.t. & Ax = 0, \\ & y_i = \delta; \quad i = 1, \ldots, n \ , \\ & (x_i, y_i, t_i) \in \mathcal{K}_i; \quad i = 1, \ldots, n \ , \\ & l \leq x \leq u \ . \end{array} \tag{5.17}$$

This model is valid even for $\delta = 0$. In that case we obtain a SOC formulation of

the $l1$-CTA (5.9)

$$\begin{aligned} \min_{x} \quad & \textstyle\sum_{i=1}^{n} w_i t_i \\ s.t. \quad & Ax = 0, \\ & (x_i, t_i) \in \mathcal{K}_i; \quad i = 1, \ldots, n \ , \\ & l \leq x \leq u \ . \end{aligned} \tag{5.18}$$

This model could have been obtained directly from $l1$-CTA (5.9) because the absolute value has an obvious second-order cone representation since the epigraph of the absolute value function is exactly second-order cone, that is,

$$t_i = |x_i| \quad \longrightarrow \quad \mathcal{K}_i = \left\{ (x_i, t_i) \in \mathbb{R}^2 \ : \ t_i \geq \sqrt{x_i^2} \right\}.$$

It is well known that the solutions of SOC problems (5.17) and (5.18) achieve solutions at the boundary of the cones, hence, equations (5.16) will hold at the solution [46, 47]. Thus, it is not necessary to enforce these equations in SOC models; in fact, their inclusion would lead to noncovex problems that would be difficult to solve.

An IPM for SOC can now be used to find an $\epsilon$-approximate solutions to SOC Pseudo-Huber and $\ell_1$ CTA models. We have used MOSEK SOC solver [45] that is considered one of the best, if not the best, SOC solver available on the market today.

Since we can solve $\ell_1$ CTA directly with SOC formulation, for practical purposes it is not necessary to consider SOC of Pseudo-Huber CTA. We included it here for theoretical purposes. However, most numerical results in next chapter do not inclide SOC of Pseudo-Huber.

# CHAPTER 6

# NUMERICAL RESULTS

In this chapter we are going to introduce the numerical results of implementation of CTA model. All results were obtained using MatLab and integrated optimizer tool MOSEK. All codes were executed on DEL OptiPlex 7040 computer with Intel(R) CORE i7-3740QM 2.70GHz processor.

## 6.1 Introductory Example

In this section we provide an example of the small two-dimensional table, that is listed in Figure 1 below as the table (a).

The continuous CTA model based on the table (a) is formulated in the following way:

- The linear constraints are obtained from the requirement that the sum of the elements in each row (or column) remains constant and is equal to the corresponding component in the last column (or row) of table (a).

- The sensitive cells are cells $a_1$ and $a_{12}$. For both of them the upper safe values are enforced, which are listed in the parentheses in the lower right corners of the cells, $upl_1 = 3$ and $upl_{12} = 5$ respectively. Hence, in the transformed tables the upper safe value of the cell $a_1$ should be 13 or above and for $a_{12}$ the upper safe value should be 18 or above.

- For the nonsensitive cells the lower and upper bounds are set to be zero and positive infinity respectively, that is, $l_{a_i} = 0$ and $u_{a_i} = \inf$ for $i = 2, \ldots, 11$.

- The weights in the objective function are set to have the value one, that is, $w_i = 1$ for $i = 1, \ldots, 12$.

From this basic CTA model different CTA models discussed in the paper were formulated and then these models were solved using appropriate IPM solvers. The results are listed in Figure 6.1.

$a$

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

(a)

LP $\ell_1$

| | | | | |
|---|---|---|---|---|
| 13 | 15 | 11 | 6 | 45 |
| 10 | 10 | 12 | 13 | 45 |
| 5 | 12 | 11 | 18 | 46 |
| 28 | 37 | 34 | 37 | 136 |

(b)

$\ell_2$

| | | | | |
|---|---|---|---|---|
| 13 | 15.03 | 11.03 | 5.94 | 45 |
| 7.66 | 11.14 | 13.14 | 13.06 | 45 |
| 7.34 | 10.83 | 9.83 | 18 | 46 |
| 28 | 37 | 34 | 37 | 136 |

(c)

SOC $\ell_1$

| | | | | |
|---|---|---|---|---|
| 13.47 | 15.26 | 11.22 | 5.05 | 45 |
| 8.19 | 10.43 | 12.43 | 13.95 | 45 |
| 6.34 | 11.31 | 10.35 | 18 | 46 |
| 28 | 37 | 34 | 37 | 136 |

(d)

SOC $\phi_{\delta=0.001}$

| | | | | |
|---|---|---|---|---|
| 13.03 | 15.39 | 11.39 | 5.19 | 45 |
| 8.37 | 10.41 | 12.41 | 13.81 | 45 |
| 6.60 | 11.20 | 10.20 | 18 | 46 |
| 28 | 37 | 34 | 37 | 136 |

(e)

Figure 6.1: Results of the small example (rounded to two decimal places).

Below is the summary of the IPM solvers used.

1. The LP-$\ell_1$-CTA (5.10) was solved using MOSEK LP solver. The IPM solver

with crossover option was used. Table (b).

2. The $\ell_2$-CTA (5.8) was solved using IPM based MOSEK QP solver. Table (c).

3. The SOC $\ell_1$-CTA (5.18) was solved using IPM based MOSEK SOC solver. Table (d).

4. The SOC Pseudo-Huber-CTA (5.17) was solved using IPM based MOSEK SOC solver. Table (e).

From Table 6.1 we can observe that SOC versions are comparable to the $\ell_2$ version both in number of iterations and CPU time; SOC $\ell_1$ was slightly faster than $\ell_2$ while SOC Pseudo-Huber was slightly slower, which is the expected result. Hence, the SOC models are more effective than the LP $\ell_1$ and Pseudo-Huber-CTA models for this example.

| CTA Model | Obj. Funct. | It. No. | CPU |
|---|---|---|---|
| LP-$\ell_1$ | 20 | 6 | 0.08 |
| $\ell_2$ | 20.69 | 6 | 0.05 |
| SOC-$\ell_1$ | 20 | 7 | 0.03 |
| SOC Pseudo-Huber | 20 | 9 | 0.06 |

Table 6.1: Results for CTA Models

Furthermore, for LP $\ell_1$, Pseudo-Huber $\phi_{0.001}$, SOC $\ell_1$, and SOC Pseudo-Huber $\phi_{0.001}$ CTA instances the optimal values of their respective objective functions are the same, namely, the value is 20, while for $\ell_2$-CTA instance it is 20.69. Thus, the objective values for SOC Pseudo Huber and $\ell_1$-CTA instances are the same as for the original non-SOC instances, namely 20, which was expected.

These results are in line with plenty of other evidence that it is advantageous to solve the SOC formulation of the problem by IPM, rather than using IPM to the original formulation of the problem (see for example [46, 47]). We are confident that the advantages of the SOC models will be even more visible when applied to larger tabular data sets, that will be provided in the next sections.

## 6.2 Implementation

In this section we provide the description of implementation of different models of CTA. As we mentioned before implementation was realized using MatLab. All codes are presented in the Appendix A. Below we provide a brief explanation of those codes.

### 6.2.1 Generating Tables of Data

For testing models we randomly generated 2D tables of different dimensions.The integrated function 'randi' is used to generate random value for each cell in the table. Upper and lower bounds for the safe values in sensitive cells were also randomly generated.

### 6.2.2 Solution Methods

Having the initial data we can test our models. All this models were realized using MOSEK optimizer.

For realization of $\ell_1$-LP model MOSEK's function 'msklpopt' was used, that is constructed currently for solving LP problems that was formulated as 4.1.

For realization of $\ell_2$ model QP MOSEK's function 'mskqpopt' was used, that's provided realization of QP problem was listed in 4.6.

For SOC for $\ell_1$ and Pseudo-Huber we used MOSEK's function 'mosekopt('symbcon')'.

For more detailed information how to use those function see MOSEK Guide [45].

### 6.2.3 User interface description

To make the work with different models and visualization of the CTA process User Interface is developed and is presented in Appendix B. A brief description of the interface is given below.

1. We start work by cleaning memory by using "Clear All" button.

2. Next, generate initial table. We have two options: first we can pick the already saved tables from the list menu, or generate new table from the interface providing minimal and maximal value of data, dimension of table and pushing "Generate Table" button.

3. Next, we decide how many Sensitive Cells we want to have and at what position. We also need to decide the safe values for the sensitive cells. For doing this we also have two options: to type the values manually or to generate random vectors of integers with given restrictions.

4. Finally, User Interface provides four options for solving corresponding problems: LP, QP, SOC for $\ell_2$ and SOC for Pseudo-Huber.

By choosing the one of four options a log window of solution appears.

The tables with numerical results of execution of this code for numerous test problem are presented in next section.

## 6.3 Numerical Results

We have tested all CTA models using User Interface described above. The results of applying different CTA models to 2D initial table with dimension 100x120 and different numbers of sensitive cells are presented in Table 6.2

| $100 \times 120$ | | | | | | |
|---|---|---|---|---|---|---|
| # of SC | Linear programing | | Quadratic Programing | | SOC for $\ell_1$ | |
| | # of Iter. | CPU Time | # of Iter. | CPU Time | # of Iter. | CPU Time |
| 2 | 9 | 1.51 | 15 | 0.53 | 12 | 0.49 |
| 3 | 8 | 1.28 | 15 | 0.64 | 12 | 0.61 |
| 4 | 8 | 1.69 | 15 | 0.39 | 12 | 0.37 |
| 5 | 8 | 1.66 | 14 | 0.41 | 12 | 0.40 |
| 10 | 7 | 1.56 | 14 | 0.42 | 12 | 0.39 |
| 20 | 8 | 1.22 | 13 | 0.38 | 11 | 0.39 |
| 50 | 8 | 0.42 | 13 | 0.3 | 11 | 0.31 |
| 100 | 7 | 0.39 | 13 | 0.25 | 10 | 0.21 |

Table 6.2: Testing initial table with dimension $100 \times 120$ using different numbers of sensitive cells

We can see from the Table 6.2 that QP has faster execution then LP. Furthermore, we see that SOC for $\ell_1$ is comparative to QP.

In table 6.3 we provide the results of implementation of different CTA models on the randomly generated tables of different (increasing) dimensions.

The analysis of the results basically confirms what we observed on the small introductory example in Section 6.1; SOC works faster then LP and competitively with QP. Note that we couldn't apply QP for tables with dimension greater then 120x150. This is issue with implementation of the problem, since in the formulation of the QP problem we need generation of identity matrix with dimension $m \times n$, and MatLab software cannot handle dimension large then 30,000. Removing this difficulty is a topic for the future research. Also, we can see that increasing dimension of table

results in increased CPU time, which was expected. However, the increase is not significant.

| 20 sensitive cells | | | | | | |
|---|---|---|---|---|---|---|
| Dimension | Linear programing | | Quadratic Programing | | SOC for $\ell_1$ | |
| | # of Iter. | CPU Time | # of Iter. | CPU Time | # of Iter. | CPU Time |
| 10x15 | 7 | 0.03 | 9 | 0.03 | 12 | 0.03 |
| 20x30 | 7 | 0.05 | 13 | 0.03 | 12 | 0.05 |
| 100 sensitive cells | | | | | | |
| 50x75 | 7 | 0.17 | 13 | 0.11 | 10 | 0.14 |
| 70x100 | 7 | 0.3 | 13 | 0.17 | 10 | 0.15 |
| 100x120 | 7 | 0.39 | 13 | 0.25 | 10 | 0.21 |
| 120x150 | 7 | 0.55 | 13 | 0.52 | 11 | 0.51 |
| 150x180 | 8 | 0.84 | - | - | 11 | 0.69 |
| 200x220 | 8 | 1.34 | - | - | 11 | 0.98 |
| 250x300 | 9 | 2.56 | - | - | 11 | 1.34 |
| 300x350 | 8 | 2.78 | - | - | 11 | 2.47 |

Table 6.3: Testing random initial table with different dimensions using fixed numbers of sensitive cells, with random position of sensitive cells and random value of sensitive cells

Initial numerical testing is encouraging and shows that SOC for $\ell_1$ formulation is a viable alternative to LP and QP formulations.

More numerical testing on tables of even bigger dimensions is needed and is forthcoming and will be the topic of further research.

# CHAPTER 7

# CONCLUDING REMARKS

In this thesis we consider methods of protection sensitive information in the tabular data sets. In particular, we consider Controlled Tabular Adjustment (CTA) model.

In CTA model we are trying to find a safe table that is minimally distanced from the original table according to the certain measure, and that is usually achieved using $\ell_1$ or $\ell_2$ norm. According to the given norm, CTA can be formulated as an optimization problem: Linear Programing (LP) for $\ell_1$-CTA, or Quadratic Programing (QP) for $\ell_2$-CTA.

In this thesis an alternative reformulation of $\ell_1$-CTA as a Second Order Cone (SOC) optimization problem was presented and numerical validity of the new approach was tested. Each model was solved using appropriate IPM. In recent years it has been shown that IPMs work well on conic optimization problems which was the motivation for the approach considered in this thesis.

The new approach was tested on the randomly generated two-dimensional tabular data sets and compared with LP and QP formulations. It was shown numerically, that SOC formulation compares favorably to QP and LP formulations.

-

# REFERENCES

[1] A. Hundepool, J. Domingo-Ferrer and others, *Statistical Disclosure Control*, John Wiley & Sons,Ltd (2012).

[2] Lambert D., *Measures of disclosure risk and harm*, Journal of Official Statistics 9(3), 313-331,(1993).

[3] Duncan G. and Lambert D., *Disclosure-limited data dissemination*, Journal of the American Statistical Association 81(3), 10-27, (1986)

[4] Fienberg S.E. and Makov U.E., *Confidentiality, uniqueness and disclosure limitation for categorical data*, Journal of Official Statistics 14(4), 385-397,(1998 )

[5] Skinner C.J. and Holmes D.J., *Estimating the re-identification risk per record in microdata*, Journal of Official Statistics 14, 361-372, (1998)

[6] Benedetti R. and Franconi L. *Statistical and technological solutions for controlled data dissemination.*, Pre-proceedings of New Techniques and Technologies for Statistics, vol. 1,pp. 225-232. Eurostat, Luxemburg, (1998)

[7] Elamir E.A.H. and Skinner C.J. *Record level measures of disclosure risk for survey microdata.*, Journal of Official Statistics 22(3), 525-539, (2006)

[8] Rinott Y. and Shlomo N. *A generalized negative binomial smoothing model for sample disclosure risk estimation.*, In Privacy in Statistical Databases, PSD 2006 (eds. DomingoFerrer J. and Franconi L.), vol. 4302 of Lecture Notes in Computer Science, pp. 82-93. Springer, Berlin/Heidelberg, (2006)

[9] Elliot M.J., Manning A.M., Mayes K., Gurd J. and Bane M. *SUDA: a program for detecting special uniques.*, Work session on Statistical Data Confidentiality, Geneva, 9-11 November 2005, pp. 353-362. Monographs in Official Statistics. Eurostat, Luxembourg, (2006)

[10] Mateo-Sanz J.M., Seb'e F. and Domingo-Ferrer J. *Outlier protection in continuous microdata masking.* In Privacy in Statistical Databases, PSD 2004 (eds. Domingo-Ferrer J. and Torra V.), vol. 3050 of Lecture Notes in Computer Science, pp. 201215. Springer, Berlin/Heidelberg, (2004)

[11] Truta T., Fotouhi F. and Barth-Jones D. *Global disclosure risk for microdata with continuous attributes.* In Privacy and Technologies of Identity (eds. Strandburg K. and Raicu D.S.), pp. 350363. Springer, US, (2006)

[12] Templ M. and Meindl B. *Robust statistics meets SDC: new disclosure risk measures for continuous microdata masking.* In Privacy in Statistical Databases, PSD 2008 (eds. Domingo-Ferrer J. and Saygn Y.), vol. 5262 of Lecture Notes in Computer Science, pp. 177189. Springer, Berlin/Heidelberg, (2008)

[13] Foschi F. *Risk for high dimensional business microdata.* Paper presented at the Joint UNECE/Eurostat work session on Statistical Data Condentiality, Tarragona, 2628 October (2011)

[14] Bacher J., Brand R. and Bender S. *Re-identifying register data by survey data using cluster analysis: an empirical study.* International Journal of Uncertainty, Fuzziness and Knowledge Based Systems 10(5), 589607, (2002)

[15] Ichim D. *Disclosure control of business microdata: a density based approach.* International Statistical Review 77, 196211, (2009)

[16] WinklerW.E. *Re-identication methods for masked microdata.* In Privacy in Statistical Databases, PSD 2004 (eds. Domingo-Ferrer J. and Torra V.), vol. 3050 of Lecture Notes in Computer Science, pp. 216230. Springer, Berlin/Heidelberg, (2004)

[17] Domingo-Ferrer J. and Torra V. *Disclosure risk assessment in statistical microdata protection via advanced record linkage.* Statistics and Computing 13(4), 343354, (2003)

[18] Skinner C. *Assessing disclosure risk for record linkage.* In Privacy in Statistical Databases, PSD 2008 (eds. Domingo-Ferrer J. and Saygn Y.), vol. 5262 of Lecture Notes in Computer Science, pp. 166176. Springer, Berlin/Heidelberg, (2008)

[19] Willenborg L.C.R.J. and de Waal A.G. *Statistical Disclosure Control in Practice.* Springer-Verlag, New York, (1996)

[20] Hundepool A., van de Wetering A., Ramaswamy R., de Wolf P.P., Franconi L., Brand R. and Domingo-Ferrer J. *-argus version 4.2 Software and Users Manual.* Statistics Netherlands, Voorburg NL. http://neon.vb.cbs.nl/casc/mu.htm.

[21] Templ M. *Statistical disclosure control for microdata using the R-package sdcMicro.* Transactions on Data Privacy 1(2), 6785, (2008)

[22] Duncan G.T. and Pearson R.W. *Enhancing access to microdata while protecting condentiality: prospects for the future.* Statistical Science 6, 219239, (1991)

[23] Sullivan G.R. *The Use of Added Error to Avoid Disclosure in Microdata Releases.* PhD thesis Iowa State University, (1989)

[24] Brand R. *Microdata protection through noise addition.* In Inference Control in Statistical Databases (ed. Domingo-Ferrer J.), vol. 2316 of Lecture Notes in Computer Science, pp. 97116. Springer, Berlin/Heidelberg, (2002)

[25] Defays, D. and Anwar, M.N. *Masking microdata using micro-aggregation.*, J. Off. Statist. 14, 449-461, (1998)

[26] Fischetti M. and Salazar-Gonz'alez J.J. *Models and algorithms for optimizing cell suppression problem in tabular data with linear constraints.* Journal of the American Statistical Association 95, 916928, (2000)

[27] CoxL.H. *Linear sensitivity measure sin statistical disclosure control.* Journal of Planning and Inference 5, 153164, (1981)

[28] Willenborg L.C.R.J. and de Waal A.G. *Elements of Statistical Disclosure Control.* Springer-Verlag, New York, (2001)

[29] Evans B.T.,Zayatz L. and Slanta J. *Using noise for disclosure limitation for establishment tabular data.* Journal of Official Statistics 14, 537552, (1998)

[30] Zayatz L. *New implementations of noise for tabular magnitude data, synthetic tabular frequency and microdata, and a remote microdata analysis system.* Work session on Statistical Condentiality, Manchester, 1719 December 2007, pp. 147157 Methodologies and Working papers. Eurostat, Luxembourg, (2007)

[31] Nayak T.K., Sinha B. and Zayatz L. *Statistical propertie sofmultiplicative noise masking for confidentiality protection.* Technical report, Statistical Research Division Research Report Series (Statistics 2010-05). U.S. Census Bureau, (2010)

[32] Massell P. and Funk J. *Recent developments in the use of noise for protecting magnitude data tables: balancing to improve data quality and rounding that*

*preserves protection.* Proceedings of the Research Conference of the Federal Committee on Statistical Methodology. Arlington, Virginia, (2007)

[33] Cox L.H. and Dandekar R.H. *Synthetic tabular data an alternative to complementary cell suppression.* Manuscript available on http://mysite.verizon.net/vze7w8vk/syn_tab.pdf, (2002)

[34] Castro J. and Giessing S. *Quality issues of minimum distance controlled tabular adjustment.* Paper presented at the European Conference on Quality in Survey Statistics, Cardiff, 2426 April, (2006)

[35] Castro J. and Giessing S. *Testing variants of minimum distance controlled tabular adjustment.* Work sessionon Statistical Data Condentiality,Geneva,911 November 2005, pp. 333343 Monographs in Ofcial Statistics. Eurostat, Luxembourg, (2006)

[36] Willenborg L.C.R.J. and de Waal A.G. *Elements of Statistical Disclosure Control.* Springer-Verlag, New York, (2001)

[37] Doyle P., Lane J.I., Theeuwes J.M.M. and Zayatz L. *Confidentiality, Disclosure and Data Access: Theory and Practical Application for Statistical Agencies.* Elsevier Science BV, North-Holland, Amsterdam, (2001)

[38] Dantzig GB. *Linear programming and extensions.* Princeton NJ: Princeton University Press (1963)

[39] Meggido N. *Pathways to the optimal set in linear programming.* In: Meggido N, Ed. Progress in mathematical programming: interior- point algorithms and related methods. Berlin: Springer-Verlag (1989); pp. 131-158.

[40] Goldman AJ, Tucker AW. *Theory of linear programming.* In: Kuhn HW, Tucker AW, Eds. Linear equalities and related systems. Princeton NJ: Princeton University Press (1956); pp. 53-97.

[41] Guler O, Ye Y. *Convergence behavior of interior-point algorithms.* Math Program (1993); 60: 215-28.

[42] Lesaja G. *Introducing Interior-Point Methods for Introductory Operations Research Courses and/or Linear Programming Courses* The Open Operational Research Journal, (2009), 3, 1-12.

[43] Philip E . Gill, Walter Murray, Michael A. Saunders, J. A. Tomlin, and Margaret H. Wright. *On projected newton barrier methods for linear programming and an equivalence to karmarkars projective method.* Mathematical Programming, 36(2):183209, November (1986).

[44] Guoyong Gu *Interior-PointMethodsfor SymmetricOptimization* Thomas Stieltjes Institute for Mathematics, (2009).

[45] E. D. Andersen, MOSEK solver. https://mosek.com/resources/doc, 2016.

[46] F. Alizadeh, and D. Goldfarb, *Second-order cone programming.* Math. Programming, 95(1):3–51, (2003).

[47] E. D. Andersen, C. Roos and T. Terlaky, *On implementing a primal-dual interior-point method for conic quadratic optimization.* Math. Programming, 95(2):249–277, (2003).

[48] J. Castro, *A CTA Model Based on the Huber Function.* Privacy in Statistical Databases 2014, LNCS, 8744:79–88, (2014).

[49] J. Castro, *Minimum-distance controlled perturbation methods for large-scale tabular data protection.* European Journal of Operational Research, 171:39–52, (2006).

[50] J. Castro and J. A. Gonzalez, *A fast CTA method without complicating binary decisions.* Documents of the Joint UNECE / Eurostat Work Session on Statistical Data Confidentiality, Statistics Canada, Ottawa, 1–7, (2013).

[51] J. Castro and J. A. Gonzalez, *A multiobjective LP approach for controlled tabular adjustment in statistical disclosure control.* Working paper, Department of Statistics and Operations Research, Universitat Politecnica de Catalunya, (2014).

[52] R. A. Dandekar and L. H. Cox, *Synthetic tabular Data: an alternative to complementary cell suppression.* Manuscript, Energy Information Administration, U.S. (2002).

# Appendix A

# CODE

## A.1 Main program

```
function varargout = models(varargin)
    gui_Singleton = 1;
    gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @models_OpeningFcn, ...
    'gui_OutputFcn', @models_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
    if nargin && ischar(varargin1)
    gui_State.gui_Callback = str2func(varargin1);
    end
    if nargout
    [varargout1:nargout] = gui_mainfcn(gui_State, varargin:);
    else
    gui_mainfcn(gui_State, varargin:);
    end
    function models_OpeningFcn(hObject, eventdata, handles, varargin)
    handles.output = hObject;
    guidata(hObject, handles);
    function varargout = models_OutputFcn(hObject, eventdata, handles)
    varargout1 = handles.output;
    function pushbutton2_Callback(hObject, eventdata, handles)
    clc;
```

```
set(handles.edit1,'String','')
set(handles.edit2,'String','')
set(handles.edit3,'String','')
set(handles.edit4,'String','')
set(handles.edit5,'String','')
set(handles.edit6,'String','')
set(handles.edit7,'String','')
set(handles.edit8,'String','')
set(handles.edit9,'String','')
set(handles.edit10,'String','')
clear all;
function edit5_Callback(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit6_Callback(hObject, eventdata, handles)
function edit6_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit7_Callback(hObject, eventdata, handles)
function edit7_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

```
function pushbutton3_Callback(hObject, eventdata, handles)
PSC=[str2num(get(handles.edit6,'String'))];
VSC=[str2num(get(handles.edit7,'String'))];
dlmwrite('PSC.txt', PSC)
dlmwrite('VSC.txt', VSC)
function edit8_Callback(hObject, eventdata, handles)
function edit8_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white'); end
function edit9_Callback(hObject, eventdata, handles)
function edit9_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function pushbutton4_Callback(hObject, eventdata, handles)
nsc=[str2num(get(handles.edit5,'String'))];
A=importdata('A.txt');
[m,n]=size(A);
SC=sort(RandSampNR(1,m*n,nsc));
dlmwrite('PSC.txt',SC)
minSC=[str2num(get(handles.edit8,'String'))];
maxSC=[str2num(get(handles.edit9,'String'))];
VSC=randi ( [minSC maxSC],1 ,nsc );
dlmwrite('VSC.txt', VSC)
```

```
function popupmenu1_Callback(hObject, eventdata, handles)
val = get(handles.popupmenu1,'Value');
if val==1
handles.current_data = importdata('A3x4.txt');
elseif val==2
handles.current_data = importdata('A10x15.txt');
elseif val==3
handles.current_data = importdata('A20x30.txt');
elseif val==4
handles.current_data = importdata('A50x75.txt');
elseif val==5
handles.current_data = importdata('A100x120.txt');
end;
dlmwrite('A.txt', handles.current_data)
function popupmenu1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit1_Callback(hObject, eventdata, handles)
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit2_Callback(hObject, eventdata, handles)
function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
```

```
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit3_Callback(hObject, eventdata, handles)
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function edit4_Callback(hObject, eventdata, handles)
function edit4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
function pushbutton1_Callback(hObject, eventdata, handles)
a=[str2num(get(handles.edit1,'String'))];
b=[str2num(get(handles.edit2,'String'))];
m=[str2num(get(handles.edit3,'String'))];
n=[str2num(get(handles.edit4,'String'))];
A=randi ( [a b], m ,n );
dlmwrite('A.txt', A)
function pushbutton5_Callback(hObject, eventdata, handles)
A=importdata('A.txt');
NSC=[str2num(get(handles.edit5,'String'))];
PSC=importdata('PSC.txt');
```

```
VSC=importdata('VSC.txt');
fprintf(2,'LINEAR PROGRAMIG MODEL/ n')
LP(A,NSC,PSC,VSC);
function pushbutton6_Callback(hObject, eventdata, handles)
A=importdata('A.txt');
NSC=[str2num(get(handles.edit5,'String'))];
PSC=importdata('PSC.txt');
VSC=importdata('VSC.txt');
fprintf(2,'QUADRATIC PROGRAMIG MODEL/n') QP(A,NSC,PSC,VSC);
function pushbutton9_Callback(hObject, eventdata, handles)
A=importdata('A.txt');
NSC=[str2num(get(handles.edit5,'String'))];
PSC=importdata('PSC.txt');
VSC=importdata('VSC.txt');
fprintf(2,'CONIC MODEL/n')
Conic_1(A,NSC,PSC,VSC);
function pushbutton10_Callback(hObject, eventdata, handles)
A=importdata('A.txt');
NSC=[str2num(get(handles.edit5,'String'))];
PSC=importdata('PSC.txt');
VSC=importdata('VSC.txt');
delta=[str2num(get(handles.edit10,'String'))];
fprintf(2,'CONIC FOR PSEUDO-HUBER WITH DELTA/n')
Conic_delta(A,NSC,PSC,VSC,delta);
function edit10_Callback(hObject, eventdata, handles)
function edit10_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

## A.2 LP function

```
function y=LP(A,NSC,PSC,VSC)
[m n]=size(A);
N=n*m;
AA=reshape(A',m*n,1)';
addpath 'c:/Program Files/mosek/7/toolbox/r2013a'
c = ones(1,2*N)';
a1=[ones(1,n),zeros(1,N-n),(-1)*ones(1,n),zeros(1,N-n)];
a2=[];
for i=1:m-1;
a2=[a2;zeros(1,i*n),ones(1,n),zeros(1,(N-(i+1)*n)),zeros(1,i*n),(-1)*ones(1,n),
zeros(1,(N-(i+1)*n))];
end;
a31=[];
for j=1:m
a31=[a31,[1,zeros(1,(n-1))]];
end
a32=[];
for j=1:m
a32=[a32,[-1,zeros(1,(n-1))]];
end
a3=[a31,a32];
```

```
a4=[];
for j=1:n-1
a4=[a4;zeros(1,j),a31(1:(end-j)),zeros(1,j),a32(1:(end-j))];
end;
a=[a1;a2;a3;a4];
blc=zeros(1,m+n)';
buc=zeros(1,m+n)';
blx1=zeros(1,N);
bux1=AA;
for k=1:length(PSC);
for i=1:N;
if i==PSC(k)
blx1(i)=VSC(k);
bux1(i)=0;
end;
end;
end;
blx=[blx1,zeros(1,N)];
bux=[inf*ones(1,N),bux1];
[res] = msklpopt(c,a,blc,buc,blx,bux);
sol = res.sol;
sol.itr.xx';
sol.itr.sux' ;
sol.itr.slx' ;
x=sol.bas.xx';
xplus=x(:,1:N);
```

```
xminus=x(:,N+1:end);
xx=xplus+xminus;
z=xx+AA;
```

## A.3 QP function

```
function y=QP(A,NSC,PSC,VSC)
[m n]=size(A);
N=n*m;
addpath 'c:/Program Files/mosek/7/toolbox/r2013a'
q = 2*eye(N);
c = [zeros(1,N)]';
a1=[ones(1,n),zeros(1,N-n)];
a2=[];
for i=1:m-1;
a2=[a2;zeros(1,i*n),ones(1,n),zeros(1,(N-(i+1)*n))];
end;
a3=[];
for j=1:m
a3=[a3,[1,zeros(1,(n-1))]];
end
a4=[];
for j=1:n-1
a4=[a4;zeros(1,j),a3(1:(end-j))];
end;
a=[a1;a2;a3;a4];
blc=zeros(1,m+n);
```

```
buc=zeros(1,m+n);
AA=reshape(A',m*n,1)';
blx=(-1)*AA;
for k=1:length(PSC);
for i=1:N;
if i==PSC(k)
blx(i)=VSC(k);
end;
end;
end;
bux = inf*ones(1,N);
[res] = mskqpopt(q,c,a,blc,buc,blx,bux);
x=res.sol.itr.xx';
z=x+AA;
s=sum(z);
```

## A.4 SOC FUNCTION

```
function y=Conic_1(A,NSC,PSC,VSC)
addpath 'c:/Program Files/mosek/7/toolbox/r2013a'
clear prob;
[r, res] = mosekopt('symbcon');
[m n]=size(A);
N=n*m;
a=zeros(1,N);
b=ones(1,N);
prob.c = [zeros(1,N),ones(1,N)];
```

```
a1=[ones(1,n),zeros(1,2*N-n)];
a2=[];
for i=1:m-1;
a2=[a2;zeros(1,i*n),ones(1,n),zeros(1,(2*N-(i+1)*n))];
end;
a3=[];
for j=1:m
a3=[a3,[1,zeros(1,(n-1))]];
end;
a4=[a3,zeros(1,N)];
a5=[];
for j=1:n-1
a5=[a5;zeros(1,j),a4(1:(end-j)),];
end;
prob.a = sparse([a1;a2;a4;a5]);
[d q]=size(prob.a);
prob.blc=[zeros(1,n+m)];
prob.buc = [zeros(1,n+m)];
AA=reshape(A',m*n,1)';
B1=(-1)*AA;
for k=1:length(PSC);
for i=1:N;
if i==PSC(k)
B1(i)=VSC(k);
end;
end;
```

```
end;
prob.blx=[B1,zeros(1,N)];
prob.bux = inf*ones(1,2*N);
probconessub=[];
for k=1:N
probconessub=[probconessub, 2*N-(N-k),N-(N-k)];
end;
probconessubptr=[1];
for s=1:N-1
probconessubptr=[probconessubptr,2*(s)+1];
end;
prob.cones.type=[res.symbcon.MSK_CT_QUAD*b]
prob.cones.sub = [probconessub];
prob.cones.subptr = [probconessubptr];
[r,res]=mosekopt('minimize',prob);
xvalue=res.sol.itr.xx';
avalue=[AA, zeros(1,N)];
zvalue=xvalue+avalue;
```

## A.5 SOC for Pseudo-Huber function

```
function y=Conic_delta(A,NSC,PSC,VSC,delta)
addpath 'c:/Program Files/mosek/7/toolbox/r2013a'
clear prob;
[r, res] = mosekopt('symbcon');
[n m]=size(A);
N=n*m;
```

```
a=zeros(1,N);
b=ones(1,N);
prob.c = [a,-b,b];
proba1=[ones(1,m), zeros(1,(3*N-m))];
proba2=[];
for i=1:n-1;
proba2=[proba2;zeros(1,i*m),ones(1,m),zeros(1,(3*N-(i+1)*m))];
end;
proba3=[];
for j=1:n
proba3=[proba3,[1,zeros(1,(m-1))]];
end
proba3=[proba3,zeros(1,2*N)];
proba4=[];
for j=1:m-1
proba4=[proba4;zeros(1,j),proba3(1:(end-j))];
end;
proba5= [zeros(N), eye(N),zeros(N)];
prob.a = sparse([proba1; proba2; proba3;proba4;proba5]);
[d q]=size(prob.a);
prob.blc=[zeros(1,n+m),delta*ones(1,N)];
prob.buc = [zeros(1,n+m),delta*ones(1,N)];
AA=reshape(A',m*n,1)';
B1=(-1)*AA;
for k=1:length(PSC);
for i=1:N;
```

```
if i==PSC(k)
B1(i)=VSC(k);
end;
end;
end;
prob.blx=[B1,zeros(1,2*N)];
prob.bux = inf*ones(1,3*N);
probconessub=[];
for k=1:N
probconessub=[probconessub, 3*N-(N-k),2*N-(N-k),N-(N-k)];
end;
probconessubptr=[1];
for s=1:N-1
probconessubptr=[probconessubptr,3*(s)+1];
end;
prob.cones.type=[res.symbcon.MSK_CT_QUAD*b];
prob.cones.sub = [probconessub];
prob.cones.subptr = [probconessubptr];
[r,res]=mosekopt('minimize',prob);
xvalue=res.sol.itr.xx';
avalue=[AA, zeros(2*N,1)'];
zvalue=xvalue+avalue;
```

# Appendix B

## USER INTERFACE

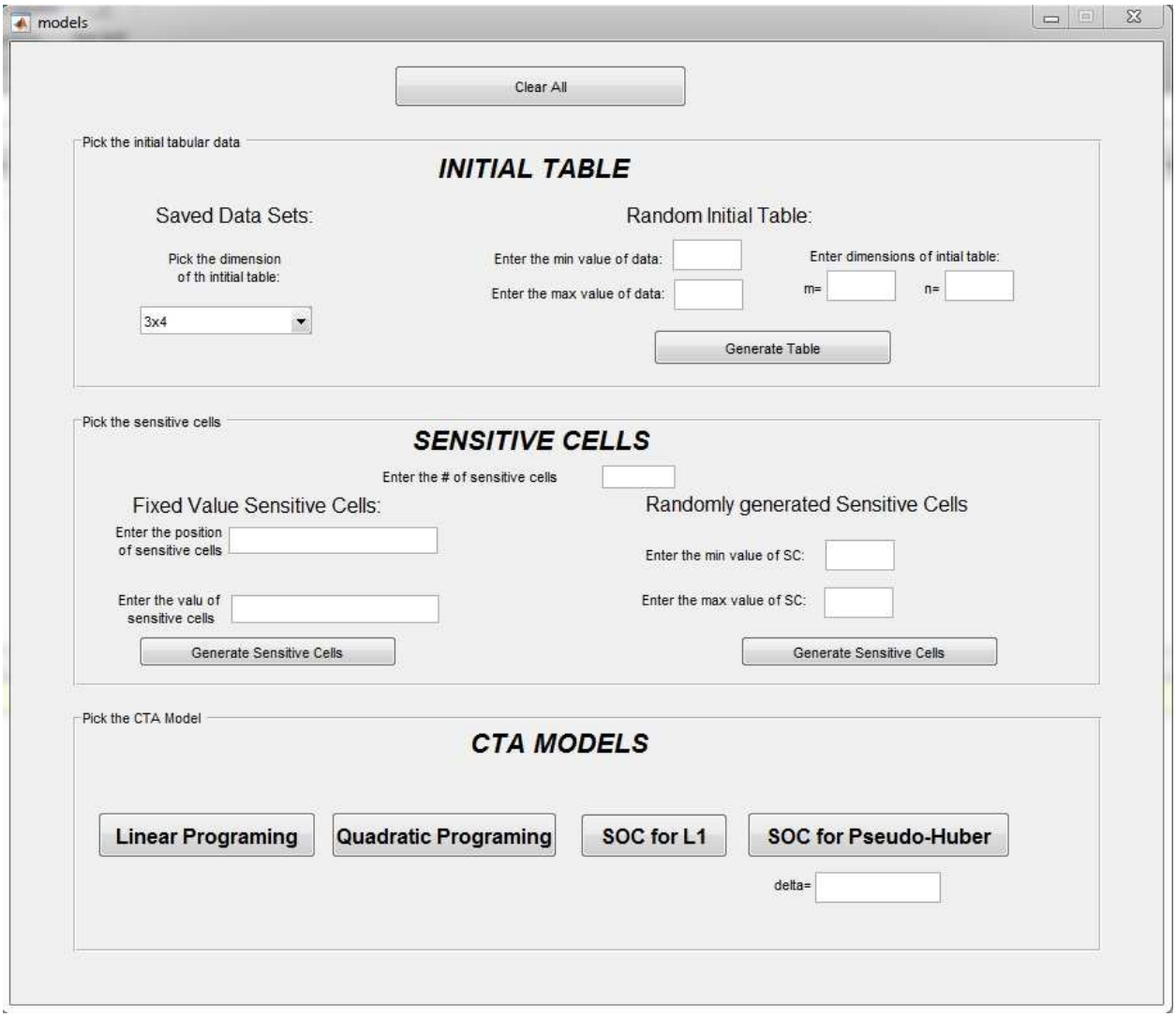


Figure B.1: User Interface