Solving L_1-CTA in 3D tables by an interior-point method for primal block-angular problems

Jordi Castro
Dept. of Stat. and Operations Research
Universitat Politècnica de Catalunya
jordi.castro@upc.edu

Jordi Cuesta
Dept. of Chemical Engineering
Universitat Rovira i Virgili
jordi.cuesta@urv.cat

Research Report UPC-DEIO DR 2011-01
March 2011

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

Noname manuscript No.
(will be inserted by the editor)

Solving L_1-CTA in 3D tables by an interior-point method for primal block-angular problems

Jordi Castro · **Jordi Cuesta**

the date of receipt and acceptance should be inserted later

Abstract The purpose of the field of statistical disclosure control is to avoid that no confidential information can be derived from statistical data released by, mainly, national statistical agencies. Controlled tabular adjustment (CTA) is an emerging technique for the protection of statistical tabular data. Given a table to be protected, CTA looks for the closest safe table. In this work we focus on CTA for three-dimensional tables using the L_1 norm for the distance between the original and protected tables. Three L_1-CTA models are presented, giving rise to six different primal block-angular structures of the constraint matrices. The resulting linear programming problems are solved by a specialized interior-point algorithm for this constraints structure, which solves the normal equations by a combination of Cholesky factorization and preconditioned conjugate gradients (PCG). In the past this algorithm shown to be one of the most efficient approaches for some classes of block-angular problems. The effect of quadratic regularizations is also analyzed, showing that for three of the six primal block-angular structures the performance of PCG is guaranteed to improve. Computational results are reported for a set of large instances, which provide linear optimization problems of up to 50 millions of variables and 25 millions of constraints. The specialized interior-point algorithm is compared with the state-of-the-art barrier solver of the CPLEX 12.1 package, showing to be a more efficient choice for very large L_1-CTA instances.

Keywords interior-point methods · primal block-angular problems · preconditioned conjugate gradient · regularizations · large-scale computational optimization · statistical tabular data protection · controlled tabular adjustment

Mathematics Subject Classification (2000) 90C90 · 90C06 · 90C08 · 90C51

Jordi Castro (✉)
Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, Jordi Girona 1–3, 08034 Barcelona, Catalonia, Spain. Tel.: +34-93-4015854. Fax.: +34-93-4015855. E-mail: jordi.castro@upc.edu, `http://www-eio.upc.es/~jcastro`

Jordi Cuesta
Statistics and Operations Research unit, Dept. of Chemical Engineering, Universitat Rovira i Virgili, Avda. Països Catalans 26, 43007 Tarragona, Catalonia, Spain. E-mail: jordi.cuesta@urv.cat

1 Introduction

Minimum-distance controlled tabular adjustment, or CTA for short, (Dandekar and Cox, 2002; Castro, 2006) is an emerging technique for the protection of statistical tabular data (Hundepool et al, 2010). This is a major concern for national statistical institutes, which must guarantee that individual information cannot be disclosed from released data. Tabular data is obtained by crossing two or more variables in a file of microdata, e.g., city, age, and profession. The Cartesian product of values for these variables provides a set of cells. For each cell, the table reports either the number of individuals (frequency or contingency tables) or information about another variable, e.g., salary (magnitude tables). Some of the state-of-the-art research in this field can be found in the recent monographs Domingo-Ferrer and Saigin (2008); Domingo-Ferrer and Magkos (2010).

Given a table to be protected, CTA looks for the closest safe table, L_1 and L_2 being the two more practical distances. We will refer as L_1-CTA and L_2-CTA to these two particular variants. CTA is formulated as a mixed integer linear or quadratic optimization problem, with binary variables. It is a difficult combinatorial optimization problem even for medium size tables, and some heuristic approaches have been recently considered for L_1-CTA (González and Castro, 2011). If the binary variables are a priori fixed, the resulting linear or quadratic optimization problems are still challenging for very large instances. In practice, interior-point methods have shown to be more efficient than simplex algorithms for linear L_1-CTA instances (Castro, 2006). As it will be outlined in the paper, the constraint matrix associated to three-dimensional (3D) tables (i.e., crossing three categorical variables) exhibits a primal block-angular structure (Castro, 2007b). In particular, the resulting L_2-CTA formulation in 3D tables is a quadratic multicommodity flow problem, and the specialized interior-point algorithm of Castro (2000) can be used. This specialized interior-point algorithm solves the normal equations at each interior-point iteration by a sensible scheme that combines Cholesky factorizations for the diagonal separable block constraints, and a preconditioned conjugate gradient (PCG) for the linking constraints. This fact was exploited in Castro (2005, 2007a); Castro and Cuesta (2011) to solve very large L_2-CTA instances. For example, some instances of 10 millions variables and 210000 constraints were solved with the specialized algorithm in 51 seconds, while CPLEX 11 needed 35000 seconds (Castro and Cuesta, 2011).

In this work we extend this approach to linear L_1-CTA in 3D tables. Due to the absolute value in the objective function, the resulting formulations are no longer multicommodity flow problems. But they still exhibit a primal block-angular structure, which can be dealt with the specialized interior-point algorithm. Three different models for L_1-CTA will be provided, each of them accepting two different primal block-angular structures. This amounts to six different primal block-angular structures. They are implemented and solved with the specialized interior-point algorithm. As it will be shown, a quadratic regularization term is instrumental to (partially) reproduce the good behaviour of the specialized algorithm for L_2-CTA. A set of 17 instances—ranging from 500 to 50 millions variables and 320 to 25 millions constraints—will be used for the computational results. It will be shown that one of the six primal block-

angular structures considered is efficient enough to outperform the state-of-the-art interior-point solver of CPLEX 12.1 in the largest instances.

The structure of the document is as follows. Section 2 outlines the formulation of the CTA problem, presents three different linear programming formulations for L_1-CTA, and shows the primal block-angular structure of CTA for 3D tables. Section 3 overviews the specialized interior-point algorithm for primal block-angular problems, and presents the main results relating the quality of the preconditioner and the presence of quadratic terms in the barrier problem (including quadratic regularizations). Section 4 presents six different primal block-angular structures for the three linear programming formulations of L_1-CTA, two for each of them; they differ in the definition of the block and linking constraints. Finally, Section 5 reports computational results with an implementation of the six different primal block-angular structures, which are solved through the specialized interior-point algorithm.

2 The L_1-CTA problem

Any CTA instance can be represented by the following parameters:

- A set of cells $a_i, i \in \mathscr{N}$, that satisfy a set linear relations $M_{j\cdot}a = b_j, j \in \mathscr{M}$. In matrix form these relations can be written $Ma = b$, $a \in \mathbb{R}^{|\mathscr{N}|}$, $M \in \mathbb{R}^{|\mathscr{M}|\times|\mathscr{N}|}$ and $b \in^{|\mathscr{M}|}$ being, respectively, the vector of a_i's, the matrix of table constraints, and the right-hand-side for these constraints.
- A lower and upper bound for each cell $i \in \mathscr{N}$, respectively $\underline{a}_i$ and $\overline{a}_i$, which are considered to be known by any attacker. If no previous knowledge is assumed for cell i $\underline{a}_i = 0$ ($\underline{a}_i = -\infty$ if $a \geq 0$ is not required) and $\overline{a}_i = +\infty$ can be used. The vectors of $\underline{a}_i$'s and $\overline{a}_i$'s are denoted as $\underline{a}$ and $\overline{a}$.
- A set $\mathscr{S} \subseteq \mathscr{N}$ of indices of sensitive or confidential cells. This cells are a priori determined by some sensitivity rules. Discussing this sensitivity rules is out of the scope of this work. Information about these rules can be found in Hundepool et al (2010).
- A lower and upper protection level for each confidential cell $i \in \mathscr{S}$, respectively lpl_i and upl_i, such that the released values $y_i, i \in \mathscr{N}$, satisfy either $y_i \geq a_i + upl_i$ or $y_i \leq a_i - lpl_i$. The vector of y_i's is denoted as y.

CTA aims at finding the closest safe values $y_i, i \in \mathscr{N}$, according to some distance L, that makes the released table safe. This involves the solution of the following optimization problem:

$$\begin{array}{ll} \min\limits_{y} & ||y-a||_L \\ \text{s. to} & My = b \\ & \underline{a} \leq y \leq \overline{a} \\ & y_i \leq a_i - lpl_i \quad \text{or} \quad y_i \geq a_i + upl_i \quad i \in \mathscr{S}. \end{array} \tag{1}$$

Problem (1) can also be formulated in terms of deviations from the current cell values. Defining $x = y - a$, $\underline{x} = \underline{a} - a$, and $\overline{x} = \overline{a} - a$, (1) can be recast as:

$$\begin{array}{ll} \min\limits_{x} & ||x||_L \\ \text{s. to} & Mx = 0 \\ & \underline{x} \le x \le \overline{x} \\ & x_i \le -lpl_i \quad \text{or} \quad x_i \ge upl_i \quad i \in \mathscr{S}, \end{array} \tag{2}$$

$x \in \mathbb{R}^{|\mathscr{N}|}$ being the vector of deviations. Problem (2) is a combinatorial optimization problem due to the disjunctive constraints. However, if the protection senses are a priori fixed for sensitive cells (i.e., only one of the two constraints $x_i \le -lpl_i$ or $x_i \ge upl_i$ is kept and the other is removed, for $i \in \mathscr{S}$), then, using the L_1 norm, (2) can be written as the following problem

$$\begin{array}{ll} \min\limits_{x} & \sum\limits_{i \in \mathscr{N}} |x_i| \\ \text{s. to} & Mx = 0 \\ & l \le x \le u, \end{array} \tag{3}$$

where $l, u \in \mathbb{R}^{|\mathscr{N}|}$ are easily obtained from $\underline{x}$, $\overline{x}$, lpl_i and upl_i, once the protection sense has been fixed.

2.1 Three linear programming formulations for (3)

The specialized interior-point solver outlined in Section 3 considers zero lower bounds for all the variables. The next three formulations meet this requirement.

The first formulation considers the change of variable $z = x - l$, such that (3) is equivalent to

$$\begin{array}{ll} \min\limits_{z} & \sum\limits_{i \in \mathscr{N}} |z_i + l_i| \\ \text{s. to} & Mz = -Al \\ & 0 \le z \le u - l. \end{array} \tag{4}$$

Adding extra variables $t \in \mathbb{R}^{|\mathscr{N}|}$ for modeling the absolute value, we finally obtain the first model:

$$\begin{array}{ll} \min\limits_{z,t} & \sum\limits_{i \in \mathscr{N}} t_i \\ \text{s. to} & Mz = -Al \\ & t \ge z + l \\ & t \ge -(z + l) \\ & 0 \le z \le u - l, \quad t \ge 0. \end{array} \tag{5}$$

The second model is directly obtained from (3) by considering the standard change of variable when dealing with L_1 norms $x = x^+ - x^-$, such that $|x_i| = x_i^+ + x_i^-$:

$$\begin{array}{ll} \min\limits_{x^+, x^-} & \sum\limits_{i \in \mathscr{N}} (x_i^+ + x_i^-) \\ \text{s. to} & M(x^+ - x^-) = 0 \\ & x^+ - x^- \le u \\ & x^+ - x^- \ge l \\ & x^+ \ge 0, \quad x^- \ge 0. \end{array} \tag{6}$$

A third model, with less constraints but more variables, is obtained from (4) by considering the change of variable $z = x^+ - x^- - l$. The third formulation is

$$\begin{array}{ll} \min\limits_{z,x^+,x^-} & \sum\limits_{i\in\mathscr{N}} (x_i^+ + x_i^-) \\ \text{s. to} & Mz = -Ml \\ & z = x^+ - x^- - l \\ & 0 \le z \le u - l, \quad x^+ \ge 0, \quad x^- \ge 0. \end{array} \tag{7}$$

2.2 The structure of table constraints in 3D tables

Crossing three categorical variables of, respectively, $r+1$, $c+1$ and $\kappa+1$ categories, we obtain a 3D table (i.e., a cube of data). The last category of each variable corresponds to marginal values, which, in practice, want to be published and can not be perturbed. The linear relations $Mx = 0$ of a 3D table are

$$\sum_{i_1=1}^{r} x_{i_1 i_2 i_3} = 0 \quad i_2 = 1 \dots c, \;\; i_3 = 1 \dots \kappa \tag{8a}$$

$$\sum_{i_2=1}^{c} x_{i_1 i_2 i_3} = 0 \quad i_1 = 1 \dots r, \;\; i_3 = 1 \dots \kappa \tag{8b}$$

$$\sum_{i_3=1}^{\kappa} x_{i_1 i_2 i_3} = 0 \quad i_1 = 1 \dots r, \;\; i_2 = 1 \dots c. \tag{8c}$$

Matrix M can be written in a primal block-angular form by exploiting the structure of (8), as follows. Firstly, variables $x_{i_1 i_2 i_3}, i_1 = 1,\dots,r, i_2 = 1,\dots,c, i_3 = 1,\dots,\kappa$ are reordered according to i_3, i.e., $x = (x_{i_1 i_2 1}^T, \dots, x_{i_1 i_2 \kappa}^T)^T$ $i_1 = 1,\dots,r, i_2 = 1,\dots,c$. Each group for a particular i_3 contains rc variables, and it is associated to one slice of the 3D cube of data. Secondly, constraints (8a)–(8b) are set first, and ordered according to i_3. Each group for a particular i_3 contains $r+c$ constraints. The remaining rc constraints (8c) are moved to end positions. The resulting constraint matrix structure is

$$M = \begin{array}{c} \begin{array}{cccc} x_{i_1 i_2 1} & x_{i_1 i_2 2} & \dots & x_{i_1 i_2 \kappa} \end{array} \\ \left[\begin{array}{cccc} N & & & \\ & N & & \\ & & \ddots & \\ & & & N \\ I & I & \dots & I \end{array}\right] \end{array} \begin{array}{l} \text{(8a–8b) for } i_3 = 1 \\ \text{(8a–8b) for } i_3 = 2 \\ \vdots \\ \text{(8a–8b) for } i_3 = \kappa \\ \text{(8c).} \end{array} \tag{9}$$

$N \in \mathbb{R}^{m\times n}$ denotes the structure of constraints (8a–8b), where $m = r + c - 1$ and $n = rc$. $I \in \mathbb{R}^{n\times n}$ are identity matrices related to constraints (8c). In addition, it was shown that N (related to the constraints and cells of a particular slice of the cube of data) is a node-arc incidence network matrix (Castro, 2007b) (one constraint is thus redundant, and this explains that the number of rows of N is $r+c-1$ instead of $r+c$). Therefore, M has a multicommodity network structure with equality linking constraints.

It is worth noting that if the Euclidean distance L_2 is considered, the general formulation of L_1-CTA in 3D tables (3) should be replaced by

$$\begin{array}{ll} \min\limits_{x} & \sum\limits_{i \in \mathcal{N}} x_i^2 \\ \text{s. to} & Mx = 0 \\ & l \leq x \leq u. \end{array} \tag{10}$$

Problem (10) is a quadratic multicommodity flow problem, and the specialized interior-point algorithm of Section 3 has shown to be the most efficient approach for this kind of problems (Castro, 2005; Castro and Cuesta, 2011). The three models for L_1-CTA (5)–(7) developed in Subsection 2.1 are no longer multicommodity problems since they involve additional constraints. In Section 4 we will show that they can be solved by the specialized interior-point algorithm.

3 Overview of the specialized interior- point algorithm for primal block-angular problems

The specialized interior-point algorithm was introduced for multicommodity flow problems in Castro (2000) and later extended to any primal block-angular structure in Castro (2007a). It considers the very general form of a primal block-angular problem

$$\min \quad \sum_{i=0}^{k} ({c^i}^T x^i + {x^i}^T Q_i x^i) \tag{11a}$$

$$\text{s. to} \quad \begin{bmatrix} N_1 & & & \\ & N_2 & & \\ & & \ddots & \\ & & & N_k \\ L_1 & L_2 & \dots & L_k \; I \end{bmatrix} \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^k \\ x^0 \end{bmatrix} = \begin{bmatrix} b^1 \\ b^2 \\ \vdots \\ b^k \\ b^0 \end{bmatrix} \tag{11b}$$

$$0 \leq x^i \leq u^i \quad i = 0, \dots, k. \tag{11c}$$

Matrices $N_i \in \mathbb{R}^{m_i \times n_i}$ and $L_i \in \mathbb{R}^{l \times n_i}$, $i = 1, \dots, k$, respectively define the block-diagonal and linking constraints, k being the number of blocks. Vectors $x^i \in \mathbb{R}^{n_i}, i = 1, \dots, k$, are the variables for each block. $x^0 \in \mathbb{R}^l$ are the slacks of the linking constraints. $b^i \in \mathbb{R}^{m_i}, i = 1, \dots, k$, is the right-hand-side vector for each block of constraints, whereas $b^0 \in \mathbb{R}^l$ is for the linking constraints. The upper bounds for each group of variables are defined by $u^i, i = 0, \dots, k$. If needed, equality constraints may be defined with this formulation by imposing (close to) zero upper bounds on the slacks. $c^i \in \mathbb{R}^{n_i}$ and $Q_i \in \mathbb{R}^{n_i \times n_i}$, $i = 1, \dots, k$, define the linear and quadratic costs for each group of variables. We restrict our considerations to the separable case where Q_i, $i = 0, \dots, k$, are diagonal positive semidefinite matrices. Although the L_1-CTA formulations of this work are linear optimization problems, quadratic terms may be used for regularizing the algorithm (Castro and Cuesta, 2011).

The specialized interior-point algorithm solves the normal equations at each interior-point iteration. The expression of normal equations (see, for instance, Wright (1996) for full details) is

$$(A\Theta A^T)\Delta y = g \tag{12}$$

where Θ is a positive definite diagonal matrix of blocks Θ_i, $i = 0,\ldots,k$, Δy is the direction of dual variables, and g is some right-hand-side g. Exploiting the structure of A and Θ in (11) the matrix of (12) can be recast as

$$\begin{aligned} A\Theta A^T &= \left[\begin{array}{ccc|c} N_1\Theta_1 N_1^T & & & N_1\Theta_1 L_1^T \\ & \ddots & & \vdots \\ & & N_k\Theta_k N_k^T & N_k\Theta_k L_k^T \\ \hline L_1\Theta_1 N_1^T & \ldots & L_k\Theta_k N_k^T & \Theta_0 + \sum_{i=1}^k L_i\Theta_i L_i^T \end{array}\right] \\ &= \begin{bmatrix} B & C \\ C^T & D \end{bmatrix}, \end{aligned} \tag{13}$$

$B \in \mathbb{R}^{\tilde{m}\times\tilde{m}}$ ($\tilde{m} = \sum_{i=1}^k m_i$), $C \in \mathbb{R}^{\tilde{m}\times l}$ and $D \in \mathbb{R}^{l\times l}$ being the blocks of $A\Theta A^T$.

Appropriately partitioning g and Δy in (12), the normal equations can be written as

$$\begin{bmatrix} B & C \\ C^T & D \end{bmatrix} \begin{bmatrix} \Delta y_1 \\ \Delta y_2 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}. \tag{14}$$

By eliminating Δy_1 from the first group of equations of (14), we obtain

$$(D - C^T B^{-1} C)\Delta y_2 = (g_2 - C^T B^{-1} g_1) \tag{15}$$

$$B\Delta y_1 = (g_1 - C\Delta y_2). \tag{16}$$

System (16) is solved by a Cholesky factorization for each diagonal block $N_i\Theta_i N_i^T, i = 1\ldots k$, of B. The system with matrix $D - C^T B^{-1} C$, the Schur complement of (14), is solved by a PCG. The dimension of this system is l, which is the number of linking constraints. In Castro (2000) it was proved that, under some conditions, which are guaranteed in our setting, the inverse of $(D - C^T B^{-1} C)$ can be computed as

$$(D - C^T B^{-1} C)^{-1} = \left(\sum_{i=0}^{\infty} (D^{-1}(C^T B^{-1} C))^i\right) D^{-1}. \tag{17}$$

The preconditioner M^{-1}, an approximation of $(D - C^T B^{-1} C)^{-1}$, is thus obtained by truncating the infinite power series (17) at some term h. The more the terms included, the better the preconditioner will be, at the expense of increasing the execution time of each PCG iteration. However, in general, $h = 0$ or $h = 1$ are reasonable choices, which in practice yield

$$\begin{aligned} M^{-1} &= D^{-1} && \text{if } h = 0, \\ M^{-1} &= (I + D^{-1}(C^T B^{-1} C))D^{-1} && \text{if } h = 1. \end{aligned}$$

This preconditioner, initially developed for multicommodity flows (Castro, 2000) can be applied to any primal block-angular problem (Castro, 2007a).

3.1 Improving the effectiveness of the preconditioner

The effectiveness of the preconditioner depends on the spectral radius of matrix $D^{-1}(C^TB^{-1}C)$, which is always in $[0,1)$ (Castro, 2000, Theorem 1). The farther away from 1 is the spectral radius of $D^{-1}(C^TB^{-1}C)$, the better is the quality of the approximation of (17) obtained by truncation with $h=0$ or $h=1$. The next theorem and proposition from Castro and Cuesta (2011) show that a quadratic term in the objective function effectively reduces this spectral radius.

Theorem 1 *Let A be the constraint matrix of problem (11), with full row rank matrices $N_i \in \mathbb{R}^{m_i \times n_i}$ $i=1,\ldots,k$, and at least one full row rank matrix $L_i \in \mathbb{R}^{l \times n_i}$, $i=1,\ldots,k$. Let Θ be the diagonal positive definite matrix of (12), and $B \in \mathbb{R}^{\tilde{m}\times\tilde{m}}$, $C \in \mathbb{R}^{\tilde{m}\times l}$ and $D \in \mathbb{R}^{l\times l}$ the submatrices of $A\Theta A^T$ defined in (13). Then, the spectral radius ρ of $D^{-1}(C^TB^{-1}C)$ is bounded by*

$$0 \le \rho \le \max_{j\in\{1,\ldots,l\}} \frac{\gamma_j}{\left(\frac{u_j}{v_j}\right)^2 \Theta_{0j} + \gamma_j} < 1, \tag{18}$$

where u is the eigenvector (or one of the eigenvectors) of $D^{-1}(C^TB^{-1}C)$ for ρ; γ_j, $j=1,\ldots,l$, and $V=[V_1 \ldots V_l]$, are respectively the eigenvalues and matrix of columnwise eigenvectors of $\sum_{i=1}^{k} L_i\Theta_i{L_i}^T$; $v=V^Tu$; and, abusing of notation, we assume that for $v_j=0$, $(u_j/v_j)^2=+\infty$.

Proposition 1 *Let assume the hypotheses of Theorem 1, and consider a linear problem and a quadratic one obtained by adding (likely small) quadratic costs $Q_i \succ 0$, $i=1,\ldots,k$. Assume $\hat{u}_j/\hat{v}_j \le u_j/v_j$, $j=1,\ldots,l$, where "hatted" and "non-hatted" terms refer, respectively, to the linear and quadratic problems, and u and v are defined as in Theorem 1. Then bound (18) is smaller for the quadratic than for the linear problem.*

The technical assumption $\hat{u}_j/\hat{v}_j \le u_j/v_j$, $j=1,\ldots,l$ in Proposition 1 is needed to guarantee that the bound (18) is smaller if a quadratic term is added to a linear problem. The fulfillment of this assumption is problem dependent, and, for a general problem, it may not be easy to check. However, as it will be shown in Section 5, matrices $L_i, i=1,\ldots,k$, of some formulations of L_1-CTA satisfy this requirement. Therefore, it makes sense to consider a quadratic regularization term in the objective function to reduce the spectral radius, i.e., to improve the quality of the preconditioner. In this work we will consider a quadratic regularization in the barrier function of the interior-point algorithm. The particular form of this regularization term is

$$R(x) = \mu\frac{1}{2}x^TQx \qquad Q = t\delta/\mu_0 I, \tag{19}$$

where x is the vector of variables, $\mu \in \mathbb{R}$ is the parameter of the barrier function, $t \in \mathbb{Z}$ is the number of interior-point iteration, $\mu_0 \in \mathbb{R}$ the value of the barrier parameter at the first interior-point iteration, and $\delta \in \mathbb{R}$ a parameter to be provided by the user for initializing the regularization matrix Q at the first iteration. Note that at the first iteration the value of the regularization is $R(x) = \mu_0 x^T(1\cdot\delta/\mu_0 I)x = \delta x^Tx$, and as we approach the optimal solution $\mu \to 0$, thus, $R(x) \to 0$. Full details about this regularized version of the algorithm can be found in Castro and Cuesta (2011).

4 Primal block-angular structures for L_1-CTA in 3D tables

Let us consider that the vectors of variables $z,t,x^+,x^- \in \mathbb{R}^{\kappa n}$ and bounds $l,u \in \mathbb{R}^{\kappa n}$, which intervene in the definition of models (5)–(7), are partitioned in κ blocks, one for each category of the third dimension, i.e., $z = ((z^1)^T,\ldots,(z^\kappa)^T)^T$, and similarly for the other vectors. Each block z^i, t^i, $(x^+)^i$, $(x^-)^i \in \mathbb{R}^n$, $i = 1,\ldots,\kappa$ is related to the variables and bounds for a particular slice of the 3D cube of data, which contains $n = rc$ cells and $m = r+c-1$ table constraints.

Exploiting the above partitioning of vectors and the structure of M in (9), (5) is equivalent to

$$\min_{z,t} \quad \sum_{i=1}^{\kappa} e^T t^i \tag{20a}$$

$$\text{s. to} \quad Nz^i = -Nl^i \qquad i = 1,\ldots,\kappa \tag{20b}$$

$$\sum_{i=1}^{\kappa} z^i = -\sum_{i=1}^{\kappa} l^i \tag{20c}$$

$$z_i - t^i \le -l^i \qquad i = 1,\ldots,\kappa \tag{20d}$$

$$-z_i - t^i \le l^i \qquad i = 1,\ldots,\kappa \tag{20e}$$

$$0 \le z^i \le u^i - l^i, \quad t^i \ge 0 \qquad i = 1,\ldots,\kappa, \tag{20f}$$

where $e \in \mathbb{R}^n$ is a vector of 1's. The constraints matrix A defined by (20b)–(20e) accept different structures, depending on the reordering of variables and constraints considered. We will consider the following two:

$$\begin{array}{c|cccccc}
 & z^1 & \cdots & z^\kappa & t^1 & \cdots & t^\kappa \\ \hline
 & N & & & & & \\
 & & \ddots & & & & \\
 & & & N & & & \\
 & I & \cdots & I & & & \\
 & I & & & -I & & \\
 & & \ddots & & & \ddots & \\
 & & & I & & & -I \\
 & -I & & & -I & & \\
 & & \ddots & & & \ddots & \\
 & & & -I & & & -I
\end{array} \tag{21}$$

$$\begin{array}{c|cccccccc}
 & z^1 & t^1 & v^1 & w^1 & \cdots & z^\kappa & t^\kappa & v^\kappa & w^\kappa \\ \hline
 & N & & & & & & & & \\
 & I & -I & I & & & & & & \\
 & -I & -I & & I & & & & & \\
 & & & & & \ddots & & & & \\
 & & & & & & N & & & \\
 & & & & & & I & -I & I & \\
 & & & & & & -I & -I & & I \\
 & I & & & & \cdots & I & & &
\end{array}, \tag{22}$$

where $v^i, w^i, i = 1,\ldots,\kappa$ are, respectively, the slacks of (20d) and (20e), and $I \in \mathbb{R}^{n\times n}$ are identity matrices. The structure of (21) is obtained with the natural ordering of (20b)–(20e). It matches the primal block-angular structure of (11) by setting $k = 2\kappa$, $N_i = N, m_i = m, n_i = n$ for $i = 1,\ldots,\kappa$, $N_i \in \mathbb{R}^{0\times n}$ for $i = \kappa+1,\ldots,2\kappa$ (empty matrix); $L_i \in \mathbb{R}^{l\times n}, i = 1,\ldots,2\kappa$ are defined by the blocks of (21) with matrices I and $-I$ for each z^i and t^i, where $l = (2\kappa+1)n$. In this scheme the block constraints are small, while the number of linking constraints is very large. The structure of (22) is obtained after reordering the variables and constraints, and transforming (20d) and

(20e) to equalities. This structure also matches (11), where $k = \kappa$,

$$N_i = \begin{bmatrix} N & & \\ I & -I & I \\ -I & -I & \end{bmatrix}, \qquad L_i = [I \;\; 0 \;\; 0 \;\; 0], \quad i = 1,\ldots,k, \tag{23}$$

$0 \in \mathbb{R}^{n\times n}$ is a matrix of 0's, $m_i = m+2n$, $n_i = 4n$, for $i = 1,\ldots,k$, and $l = n$.

Similarly, exploiting the structure of M, (6) is equivalent to

$$\begin{aligned}
\min_{x^+,x^-} \quad & \sum_{i=1}^{\kappa} e^T(x^{+i} + x^{-i}) && && \text{(24a)}\\
\text{s. to} \quad & N(x^{+i} - x^{-i}) = 0 && i = 1,\ldots,\kappa && \text{(24b)}\\
& \sum_{i=1}^{\kappa} (x^{+i} - x^{-i}) = 0 && && \text{(24c)}\\
& x^{+i} - x^{-i} \le u^i && i = 1,\ldots,\kappa && \text{(24d)}\\
& -x^{+i} + x^{-i} \le -l^i && i = 1,\ldots,\kappa && \text{(24e)}\\
& x^{+i} \ge 0, \quad x^{-i} \ge 0 && i = 1,\ldots,\kappa. && \text{(24f)}
\end{aligned}$$

The two particular structures, with and without reordering, of the constraints matrix A defined by (24b)–(24e) are

$$\begin{array}{c}
\begin{array}{cccccc} x^{+1} & x^{-1} & \ldots & x^{+\kappa} & x^{-\kappa} \end{array} \\
\left[\begin{array}{cccccc}
N & -N & & & & \\
 & & \ddots & & & \\
 & & & N & -N & \\
I & -I & \cdots & I & -I & \\
I & -I & & & & \\
 & & \cdots & & & \\
 & & & I & -I & \\
-I & I & & & & \\
 & & \cdots & & & \\
 & & & -I & I &
\end{array}\right]
\end{array} \tag{25}$$

$$\begin{array}{c}
\begin{array}{cccccccccc} x^{+1} & x^{-1} & v^1 & w^1 & \ldots & x^{+\kappa} & x^{-\kappa} & v^\kappa & w^\kappa \end{array} \\
\left[\begin{array}{cccccccccc}
N & -N & & & & & & & & \\
I & -I & I & & & & & & & \\
-I & I & & I & & & & & & \\
 & & & & \ddots & & & & & \\
 & & & & & N & -N & & & \\
 & & & & & I & -I & I & & \\
 & & & & & -I & I & & I & \\
I & -I & & & \cdots & I & -I & & &
\end{array}\right]
\end{array} \tag{26}$$

where $v^i, w^i, i = 1,\ldots,\kappa$ are, respectively, the slacks of (24d) and (24e). Both (25) and (26) match (11). For (25) we have $k = \kappa$, $N_i = [N \;\; -N]$, $m_i = m$, $n_i = 2n$, $l = (2\kappa+1)n$, and $L_i \in \mathbb{R}^{l\times 2n}, i = 1,\ldots,\kappa$ are defined by the blocks of (25) with matrices I and $-I$ for (x^{+i}, x^{-i}). On the other hand, for (26), $k = \kappa$, $m_i = m+2n$, $n_i = 4n$, $l = n$, and

$$N_i = \begin{bmatrix} N & -N & \\ I & -I & I \\ -I & I & I \end{bmatrix}, \qquad L_i = [I \;\; -I \;\; 0 \;\; 0], \quad i = 1,\ldots,k,. \tag{27}$$

Finally, (7) is equivalent to

$$\min_{z,x^+,x^-} \quad \sum_{i=1}^{\kappa} e^T(x^{+i}+x^{-i}) \tag{28a}$$

$$\text{s. to} \quad Nz^i = -Nl^i \qquad i=1,\dots,\kappa \tag{28b}$$

$$\sum_{i=1}^{\kappa} z^i = -\sum_{i=1}^{\kappa} l^i \tag{28c}$$

$$z^i - x^{+i} + x^{-i} = -l^i \qquad i=1,\dots,\kappa \tag{28d}$$

$$0 \le z^i \le u^i - l^i, \quad x^{+i} \ge 0, \quad x^{-i} \ge 0 \qquad i=1,\dots,\kappa. \tag{28e}$$

The two particular structures, with and without reordering, of the constraints matrix A defined by (28b)–(28d) are

$$\begin{array}{c} z^1 \ \dots \ z^\kappa \ x^{+1} \ x^{-1} \ \dots \ x^{+\kappa} \ x^{-\kappa} \\ \left[\begin{array}{cccccccc} N & & & & & & & \\ & \ddots & & & & & & \\ & & N & & & & & \\ I & \cdots & I & & & & & \\ I & & & -I & I & & & \\ & \ddots & & & & \ddots & & \\ & & I & & & & -I & I \end{array}\right] \end{array} \tag{29}$$

$$\begin{array}{c} z^1 \ x^{+1} \ x^{-1} \ \dots \ z^\kappa \ x^{+\kappa} \ x^{-\kappa} \\ \left[\begin{array}{cccccccc} N & & & & & & & \\ I & -I & I & & & & & \\ & & & \ddots & & & & \\ & & & & N & & & \\ & & & & I & -I & I & \\ I & & & \cdots & I & & & \end{array}\right] \end{array} . \tag{30}$$

As for previous models, (29) and (30) match (11). For structure (29) $k = 2\kappa$, $N_i = N, m_i = m, n_i = n$ for $i = 1,\dots,\kappa$, $N_i \in \mathbb{R}^{0\times 2n}$ for $i = \kappa+1,\dots,2\kappa$ (empty matrix), $l = (\kappa+1)n$, $L_i \in \mathbb{R}^{l\times n}$ for $i = 1,\dots,\kappa$, and $L_i \in \mathbb{R}^{l\times 2n}$ for $i = \kappa+1,\dots,2\kappa$. For (30), $k = \kappa$, $m_i = m+n$, $n_i = 3n$, $l = n$, and

$$N_i = \begin{bmatrix} N & & \\ I & -I & I \end{bmatrix}, \qquad L_i = [I \ \ 0 \ \ 0], \quad i = 1,\dots,k,. \tag{31}$$

The three pairs of structures (21)–(22), (25)–(26) and (29)–(30), amount to six different formulations of L_1-CTA which can be solved with the specialized algorithm of Section 3. Formulations (21), (25), (29) have in common very small block diagonal matrices N_i, but a large number l of linking constraints. On the other hand, structures (22), (26) and (30) have larger diagonal matrices N_i but significantly fewer linking constraints. A priori the latter seem to be more promising approaches, since the fewer linking constraints, the higher the performance of the specialized algorithm. The computational comparison made at the beginning of Section 5 between the two groups of formulations confirms this guess. Another advantage of formulations (22), (26) and (30) is that the topology of N_i and L_i is the same for all the blocks $i = 1,\dots,k$, which is a slight computational benefit when performing the symbolic factorizations in the interior-point algorithm. However, the most instrumental feature is that, as shown by below Proposition 2, the structures of matrices L_i for (22), (26) and (30) satisfy the technical assumption $\hat{u}_j/\hat{v}_j \le u_j/v_j$, $j = 1,\dots,l$, of Proposition 1. This guarantees that quadratic terms in the objective function (which can be added by a quadratic regularization (Castro and Cuesta, 2011)) will improve the performance of the algorithm.

Proposition 2 *Given the hypotheses of Theorem 1, let us consider the definition of u and v in that Theorem. Let u, v be the vectors $\hat{u}$ and $\hat{v}$ for a quadratic problem obtained by adding a quadratic term to the linear objective. Then, for the constraint matrices A defined by* (22)*,* (26) *and* (30)*, it holds that $\hat{u}_j/\hat{v}_j = u_j/v_j,\ j = 1,\ldots,l$.*

Proof By definition, $u, v \in \mathbb{R}^l$, $v = V^T u$, where $V = [V_1 \ldots V_l] \in \mathbb{R}^{l\times l}$, is the matrix of columnwise eigenvectors $V_j, j = 1,\ldots,l$ of $\sum_{i=1}^{k} L_i \Theta_i {L_i}^T$. Matrices $L_i, i = 1,\ldots,k$ for (22), (26) and (30) are, respectively, defined by (23), (27), (31). The matrices $\sum_{i=1}^{k} L_i \Theta_i {L_i}^T$ derived from (23), (27), (31) are thus, respectively, $\sum_{i=1}^{k} \Theta_{z^i}$, $\sum_{i=1}^{k} (\Theta_{x^{+i}} - \Theta_{x^{-i}})$ and $\sum_{i=1}^{k} \Theta_{z^i}$, which are positive definite and diagonal. Therefore $V = I$, $v = Iu = u$, and similarly for the linear objective $\hat{V} = I$, $\hat{v} = I\hat{u} = \hat{u}$, thus satisfying $\hat{u}_j/\hat{v}_j = u_j/v_j = 1$, $j = 1,\ldots,l$. □

By Propositions 2 and 1 the bound of the spectral radius (18) is reduced if a quadratic regularization is added to the objective function. A it will be shown in Section 5, the regularized version of the specialized algorithm is by far more efficient than the non regularized version for large L_1-CTA instances, making it competitive against state-of-the-art interior-point implementations like CPLEX 12.1.

5 Computational results

The six different formulations (21)–(22), (25)–(26) and (29)–(30) have been implemented and solved with the specialized interior-point algorithm. We used a MATLAB implementation of the algorithm described in Castro (2007a), named PRBLOCK_IP. PRBLOCK_IP implements a standard infeasible path-following algorithm (Wright, 1996), which solves normal equations either through a Cholesky factorization, or through the specialized procedure. Note that neither the Mehrotra predictor-corrector (Mehrotra, 1992) nor the multiple centrality corrections (Gondzio, 1996) heuristics for higher order directions are not used, since, as shown in Castro (2000), they are not useful when PCG is applied in interior-point methods. For reasons of efficiency, Cholesky factorizations are performed through external precompiled routines. Specifically, the code uses the Ng-Peyton sparse Cholesky package Ng and Peyton (1993), hooked to MATLAB for the LIPSOL package Zhang (1998). The code also includes the regularization strategy described in Castro and Cuesta (2011). PRBLOCK_IP can be obtained for research purposes from `http://www-eio.upc.es/~jcastro/prblock_ip.html`. Results with the barrier algorithm of CPLEX 12.1 are also reported. The CPLEX simplex solvers were not used, since interior-point algorithms have shown to be the most efficient option for the linear programming formulations of CTA (Castro, 2006). MATLAB and CPLEX were hooked by a free software, available from `http://www-eio.upc.es/~jcastro/software.html`. It is worth noting that CPLEX 12.1 is a state-of-the-art code with highly optimized routines, while the Ng-Peyton package used by PRBLOCK_IP is an academic free implementation for sparse Cholesky factorizations. Therefore, the comparison between PRBLOCK_IP and CPLEX 12.1 is biased due to the quality of the implementation. All runs were carried out on a Dell PowerEdge 6950 server with four dual core AMD Opteron 8222 3.0 GHZ processors and 64 GB of RAM, without exploitation of parallelism capabilities.

Table 1 Dimensions of L_1-CTA instances for formulations (21)–(22)

	Formulation (21)			Formulation (22)				
Instance	k	$m_i^{(*)}$	n_i	k	m_i	n_i	vars.	const.
CTA-5-5-5	10	9	25	5	59	100	500	320
CTA-10-10-5	10	19	100	5	219	400	2000	1195
CTA-15-15-10	20	29	225	10	479	900	9000	5015
CTA-20-20-20	40	39	400	20	839	1600	32000	17180
CTA-25-25-10	20	49	625	10	1299	2500	25000	13615
CTA-25-25-25	50	49	625	25	1299	2500	62500	33100
CTA-30-30-10	20	59	900	10	1859	3600	36000	19490
CTA-30-30-30	60	59	900	30	1859	3600	108000	56670
CTA-40-40-20	40	79	1600	20	3279	6400	128000	67180
CTA-50-50-10	20	99	2500	10	5099	10000	100000	53490
CTA-50-50-25	50	99	2500	25	5099	10000	250000	129975
CTA-50-50-50	100	99	2500	50	5099	10000	500000	257450
CTA-100-100-10	20	199	10000	10	20199	40000	400000	211990
CTA-100-100-25	50	199	10000	25	20199	40000	1000000	514975
CTA-200-200-50	100	399	40000	50	80399	160000	8000000	4059950
CTA-500-500-50	100	999	250000	50	500999	1000000	50000000	25299950
CTA-1000-500-20	40	1499	500000	20	1001499	2000000	40000000	20529980

$^{(*)}$ Only for first κ nonempty blocks

As MATLAB is an interpreted language, the overall execution time is meaningless. Following Castro (2007a), we only report the execution time spent in the external precompiled Ng-Peyton Cholesky routines, including minimum degree ordering, symbolic factorization, numerical factorization, and numerical solution. Note that communication between MATLAB and the external precompiled Ng-Peyton Cholesky routines is made *b*y value (copy of parameters). This means a significant overhead time for large instances, which is included in the time reported. For large instances, this reported time should be comparable to the total time spent by an efficient C/C++ implementation. For the runs with CPLEX 12.1 we provide the overall execution time. An efficient C/C++ version of PRBLOCK_IP is work in progress.

We generated a set of 17 3D CTA instances with a random generator of synthetic tables. The generator can be retrieved from `http://www-eio.upc.es/~jcastro/CTA_3Dtables.html`. Tables 1–3 report the dimensions of each instance, for, respectively, the formulations (21)–(22), (25)–(26) and (29)–(30). Instances are denoted as CTA-c-r-κ, where r, c and κ are the number of categories of the variables of the 3D table. For each instance and formulation the tables report the number of blocks (k), the number of constraints and variables for each block (m_i and n_i), and the overall number of constraints and variables of the linear problem ("vars." and "const."), including the slacks of inequality linking constraints. For formulations (21) and (29) columns m_i provide the number of constraints of the first κ nonempty diagonal blocks.

From tables 1–3 it is clear that, although the overall number of variables and constraints of the resulting linear problem is the same for the two formulations of each table, the dimension of the diagonal matrices ($m_i \times n_i$) of formulations (21), (25) and (29) is much smaller (or, equivalently, the number of linking constraints of formulations (21), (25) and (29) is much larger). Since the dimension of systems to be solved

Table 2 Dimensions of L_1-CTA instances for formulations (25)–(26)

	Formulation (25)			Formulation (26)				
Instance	k	m_i	n_i	k	m_i	n_i	vars.	const.
CTA-5-5-5	5	9	50	5	59	100	500	320
CTA-10-10-5	5	19	200	5	219	400	2000	1195
CTA-15-15-10	10	29	450	10	479	900	9000	5015
CTA-20-20-20	20	39	800	20	839	1600	32000	17180
CTA-25-25-10	10	49	1250	10	1299	2500	25000	13615
CTA-25-25-25	25	49	1250	25	1299	2500	62500	33100
CTA-30-30-10	10	59	1800	10	1859	3600	36000	19490
CTA-30-30-30	30	59	1800	30	1859	3600	108000	56670
CTA-40-40-20	20	79	3200	20	3279	6400	128000	67180
CTA-50-50-10	10	99	5000	10	5099	10000	100000	53490
CTA-50-50-25	25	99	5000	25	5099	10000	250000	129975
CTA-50-50-50	50	99	5000	50	5099	10000	500000	257450
CTA-100-100-10	10	199	20000	10	20199	40000	400000	211990
CTA-100-100-25	25	199	20000	25	20199	40000	1000000	514975
CTA-200-200-50	50	399	80000	50	80399	160000	8000000	4059950
CTA-500-500-50	50	999	500000	50	500999	1000000	50000000	25299950
CTA-1000-500-20	20	1499	1000000	20	1001499	2000000	40000000	20529980

Table 3 Dimensions of L_1-CTA instances for formulations (29)–(30)

	Formulation (29)			Formulation (30)				
Instance	k	$m_i^{(*)}$	n_i	k	m_i	n_i	vars.	const.
CTA-5-5-5	10	9	25	5	34	75	375	195
CTA-10-10-5	10	19	100	5	119	300	1500	695
CTA-15-15-10	20	29	225	10	254	675	6750	2765
CTA-20-20-20	40	39	400	20	439	1200	24000	9180
CTA-25-25-10	20	49	625	10	674	1875	18750	7365
CTA-25-25-25	50	49	625	25	674	1875	46875	17475
CTA-30-30-10	20	59	900	10	959	2700	27000	10490
CTA-30-30-30	60	59	900	30	959	2700	81000	29670
CTA-40-40-20	40	79	1600	20	1679	4800	96000	35180
CTA-50-50-10	20	99	2500	10	2599	7500	75000	28490
CTA-50-50-25	50	99	2500	25	2599	7500	187500	67475
CTA-50-50-50	100	99	2500	50	2599	7500	375000	132450
CTA-100-100-10	20	199	10000	10	10199	30000	300000	111990
CTA-100-100-25	50	199	10000	25	10199	30000	750000	264975
CTA-200-200-50	100	399	40000	50	40399	120000	6000000	2059950
CTA-500-500-50	100	999	250000	50	250999	750000	37500000	12799950
CTA-1000-500-20	40	1499	500000	20	501499	1500000	30000000	10529980

$^{(*)}$ Only for first κ nonempty blocks

by PCG in the the specialized interior-point algorithm is the number of linking constraints, (21), (25) and (29) seem not to be good candidates. This is confirmed by the results of Tables 4–6. They provide the computational results obtained with, respectively, (21)–(22), (25)–(26) and (29)–(30), for a subset of small instances. Columns "It.", "PCG" and "CPU" provide the number of interior-point iterations, overall number of PCG iteration and CPU time. The maximum allowed number of interior-point

Table 4 Results for a subset of small instances comparing (21) and (22)

	Formulation (21)			Formulation (22)		
Instance	It.	PCG	CPU	It.	PCG	CPU
CTA-5-5-5	51(9)	4670	1.38	200	280	0.34
CTA-10-10-5	48(13)	14851	4.93	200	1724	0.77
CTA-15-15-10	100(44)	145745	152.09	111(23)	5382	4.57
CTA-20-20-20	100(21)	503361	1575.34	200(118)	6807	40.24
CTA-25-25-10	98(36)	411112	854.00	93(6)	17958	27.40
CTA-25-25-25	100(20)	1105761	6939.38	200	5936	30.80
CTA-30-30-10	117(40)	672017	2475.10	200(74)	36583	77.22
CTA-30-30-30	100(4)	1458906	15883.20	200(66)	27319	343.88

Table 5 Results for a subset of small instances comparing (25) and (26)

	Formulation (25)			Formulation (26)		
Instance	It.	PCG	CPU	It.	PCG	CPU
CTA-5-5-5	18	65	0.06	27	76	0.07
CTA-10-10-5	48	1671	0.38	58	502	0.22
CTA-15-15-10	46	3247	2.92	48	1951	1.47
CTA-20-20-20	51	13722	30.22	57(15)	3528	11.40
CTA-25-25-10	64	22556	37.33	48	4594	6.79
CTA-25-25-25	48	14437	95.68	36	1454	6.17
CTA-30-30-10	54	6501	15.89	75(21)	7806	20.30
CTA-30-30-30	58	13106	161.67	70(20)	5624	90.06

Table 6 Results for a subset of small instances comparing (29) and (30)

	Formulation (29)			Formulation (30)		
Instance	It.	PCG	CPU	It.	PCG	CPU
CTA-5-5-5	54(5)	1493	0.51	154(9)	252	0.22
CTA-10-10-5	94(8)	4077	1.56	41	1033	0.30
CTA-15-15-10	141	87208	65.22	66	3685	2.33
CTA-20-20-20	200(39)	189318	423.75	200(110)	13444	35.37
CTA-25-25-10	162(23)	114437	239.17	109(6)	29714	27.83
CTA-25-25-25	200(36)	105095	472.56	200	5907	16.73
CTA-30-30-10	147	35540	57.48	187(60)	45128	61.34
CTA-30-30-30	200(21)	387852	2345.41	200(54)	29252	228.41

iterations was set to either 100 or 200, depending on the instance. Executions that reached this maximum limit provided a suboptimal solution. For columns "It.", the values in brackets correspond to the number of iterations performed solving normal equations by Cholesky factorizations, instead of PCG; the code automatically switches from PCG to Cholesky when a wrong solution is reported by PCG within its maximum number of iterations (set to a large value of $2l$). From Tables 4–6 it is clear that the specialized interior-point algorithm is more efficient with formulation (22) than with (21), mainly due the big difference in number of PCG iterations needed by the formulations. A similar conclusion can be derived for (25) and (26), the latter being a more efficient approach. In that case, however, for (26) the code switched from PCG to Cholesky, while PCG was always enough for (25) (at the expenses of a larger

Table 7 Results for a subset of instances comparing (22), (26) and (30)

	Formulation (22)			Formulation (26)			Formulation (30)		
Instance	It.	PCG	CPU	It.	PCG	CPU	It.	PCG	CPU
CTA-40-40-20	200(41)	14319	181.99	57	5256	49.89	200(61)	18303	154.59
CTA-50-50-10	200(93)	30910	232.18	54	5867	36.18	200(77)	86882	315.06
CTA-50-50-25	200(65)	15939	548.88	200(89)	13554	564.70	200(74)	23816	532.57
CTA-50-50-50	200(63)	124991	6779.41	140(17)	105453	4142.12	200(71)	111273	4835.68
CTA-100-100-10	200(94)	34339	1441.62	200(110)	33176	1677.41	200(109)	60610	1632.50
CTA-100-100-25	200(96)	26844	5537.92	93(32)	6849	2718.50	200(102)	38229	5346.47
CTA-200-200-50	Not executed			45	281	599.44	Not executed		

number of PCG iterations). Similarly, (30) outperformed (29). This empirical evidence confirms that those formulations with fewer linking constraints are favorable for the specialized interior-point algorithm.

Table 7 reports results for the more efficient formulations (22), (26) and (30) considering the subset of larger instances. The meaning of the columns is as in previous tables. The limit of interior-point iterations was set to 200. We see that the limit was reached in many runs. In addition, the code switched from PCG to Cholesky in almost all runs. However, when this happened, the relative optimality gap between the dual and primal objective was in general $O(10^{-5})$. The remaining —and expensive— Cholesky factorization were needed to reach the required default 10^{-6} optimality gap. Without this interior-point iterations which solve the Cholesky factorizations of normal equations, the CPU time would have been much lower. We see that formulation (26) provides the best results, both in CPU time, but also in number of PCG and interior-point iterations. Indeed, this option was the only one that solved three instances without switching to the Cholesky factorization of normal equations. We also checked that, when this switch is done, the optimality gap reached by formulation (26) was smaller than for the other two formulations.

Finally, Table 8 shows results with the regularized interior-point algorithm, for the more efficient formulations (22), (26) and (30), and with the barrier solver of CPLEX 12.1. The meaning of the columns is as for previous tables. All the instances were considered, excluding the three largest ones for the formulations (22) and (30), since the computational time would be excessive. The value $\delta = 10^{-3}$ was used for the regularization term (19). Note that for CPLEX the particular formulation considered is not relevant, since it does not exploit a particular partitioning of the constraints matrix; anyway, the results provided for CPLEX correspond to the ordering of constraints and variables of (26). From Table 8 it is clear that: (1) regularized versions are much more efficient than the non-regularized ones; (2) formulation (26) is the most efficient approach for the regularized algorithm; (3) the generic barrier of CPLEX 12.1 is the most efficient approach when the size of the problem is not very large; (4) for the tree largest instances (of millions of variables and constraints), the specialized approach outperforms the state-of-the-art interior-point solver.

Table 8 Results comparing (22), (26) and (30), with the regularized interior-point algorithm, and the barrier solver of CPLEX 12.1

	Formulation (22)			Formulation (30)		
Instance	It.	PCG	CPU	It.	PCG	CPU
CTA-5-5-5	200	280	0.27	25	82	0.07
CTA-10-10-5	29	397	0.21	157(7)	1894	0.63
CTA-15-15-10	94	7034	4.68	63	3497	1.95
CTA-20-20-20	36	2125	5.11	42	3140	4.61
CTA-25-25-10	48	5705	7.98	49	5841	5.60
CTA-25-25-25	58	2841	11.25	127	4233	12.51
CTA-30-30-10	119	38426	72.52	134	54394	64.11
CTA-30-30-30	100	14895	104.76	65	13753	55.72
CTA-40-40-20	200(39)	19865	196.53	200(68)	17893	139.35
CTA-50-50-10	200(58)	54886	344.81	200(46)	112168	389.36
CTA-50-50-25	200(43)	21654	514.96	172(58)	19365	369.13
CTA-50-50-50	200(56)	176643	7762.12	125	134386	2509.18
CTA-100-100-10	130(26)	120548	3382.39	108(7)	70431	1337.92
CTA-100-100-25	200(79)	33464	6607.60	112(8)	43507	3225.03
CTA-200-200-50		not executed			not executed	
CTA-500-500-50		not executed			not executed	
CTA-1000-500-20		not executed			not executed	

	Formulation (26)			barrier CPLEX 12.1	
Instance	It.	PCG	CPU	It.	CPU
CTA-5-5-5	28	128	0.16	7	0.01
CTA-10-10-5	71	600	0.34	9	0.01
CTA-15-15-10	32	898	0.68	15	0.07
CTA-20-20-20	35	2011	4.87	13	0.73
CTA-25-25-10	39	2405	4.19	17	0.28
CTA-25-25-25	36	1358	6.37	9	1.80
CTA-30-30-10	48	4733	10.37	18	0.51
CTA-30-30-30	54	7865	52.36	14	6.85
CTA-40-40-20	46	3301	27.39	12	4.98
CTA-50-50-10	56	6774	41.36	28	2.72
CTA-50-50-25	50	4558	73.36	13	13.64
CTA-50-50-50	113	73002	1959.40	25	166.87
CTA-100-100-10	77	13727	319.79	30	24.28
CTA-100-100-25	82	26292	1604.76	20	153.92
CTA-200-200-50	42	282	514.26	9	3518.21
CTA-500-500-50	53	163	3631.98	9	55670.34
CTA-1000-500-20	51	160	2723.17	8	10050.54

6 Conclusions

The specialized interior-point algorithm for primal block-angular problems was shown in the past to be an efficient approach for large continuous L_2-CTA instances, which result in quadratic multicommodity flow problems. However, until this work, the algorithm had not been applied to L_1-CTA. L_1-CTA has a primal block-angular structure, but it is no longer a multicommodity flow problem. However, for one of the six primal block-angular structures developed, a regularized version of the specialized interior-point algorithm has shown to be an efficient approach, outperforming the state-of-the-art interior-point solver of CPLEX 12.1 when the size of the problem is very large (millions of variables and constraints). The regularization term has shown to be instrumental in the efficiency of this approach. For larger instances, the specialized algorithm is expected to be even more efficient against CPLEX 12.1. However such massive instances can not be dealt with the current MATLAB code, and a C/C++

implementation of the specialized algorithm should be used. The development of this efficient implementation is one of the ongoing projects.

Acknowledgements This work has been supported by grants MTM2009-08747 of the Spanish Ministry of Science and Innovation, and SGR-2009-1122 of the Government of Catalonia.

References

Castro J (2000) A specialized interior-point algorithm for multicommodity network flows. SIAM J Opt. 10:852–877

Castro J (2005) Quadratic interior-point methods in statistical disclosure control. Comp Manag Sci. 2:107–121

Castro J (2006) Minimum-distance controlled perturbation methods for large-scale tabular data protection. Eur J Oper Res. 171:39–52

Castro J (2007a) An interior-point approach for primal block-angular problems, Comput Opt App. 36:195–219.

Castro J (2007b) A shortest paths heuristic for statistical disclosure control in positive tables. INFORMS J Comput. 19:520–533

Castro J, Cuesta J (2011) Quadratic regularizations in an interior-point method for primal block-angular problems. Math Prog. doi:10.1007/s10107-010-0341-2

Domingo-Ferrer J, Magkos E (eds.) (2010), *Lecture Notes in Computer Science. Privacy in Statistical Databases* (Vol. 6344). Springer, Berlin

Domingo-Ferrer J, Saigin Y (eds.) (2008), *Lecture Notes in Computer Science. Privacy in Statistical Databases* (Vol. 5262). Springer, Berlin

Dandekar RA, Cox LH (2002), Synthetic tabular data: An alternative to complementary cell suppression, manuscript, Energy Information Administration, US Department of Energy.

Gondzio J (1996) Multiple centrality corrections in a primal dual method for linear programming. Comput Opt App. 6:137–156

González JA, Castro J (2011) A heuristic block coordinate descent approach for controlled tabular adjustment. Comp Oper Res. doi:10.1016/j.cor.2011.02.008

Hundepool A, Domingo-Ferrer J, Franconi L, Giessing S, Lenz R, Naylor J, Schulte–Nordholt E, Seri E, de Wolf PP (2010), Handbook on statistical disclosure control (v 1.2). Network of Excellence in the European Statistical System in the field of Statistical Disclosure Control. `http://neon.vb.cbs.nl/casc/SDC_Handbook.pdf`

Mehrotra S (1992) On the implementation of a primal-dual interior point method. SIAM J Opt. 2: 575–601

Ng E, Peyton BW (1993) Block sparse Cholesky algorithms on advanced uniprocessor computers. SIAM J Sci Comput. 14:1034–1056

Wright SJ (1996) Primal-Dual Interior-Point Methods. SIAM, Philadelphia

Zhang Y (1998) Solving large-scale linear programs by interior-point methods under the MATLAB environment. Opt Methods Soft. 10:1–31