On implementing a primal-dual interior-point method for conic quadratic optimization

E. D. Andersen[*], C. Roos[†], and T. Terlaky[‡]

December 18, 2000

Abstract

Conic quadratic optimization is the problem of minimizing a linear function subject to the intersection of an affine set and the product of quadratic cones. The problem is a convex optimization problem and has numerous applications in engineering, economics, and other areas of science. Indeed, linear and convex quadratic optimization is a special case.

Conic quadratic optimization problems can in theory be solved efficiently using interior-point methods. In particular it has been shown by Nesterov and Todd that primal-dual interior-point methods developed for linear optimization can be generalized to the conic quadratic case while maintaining their efficiency. Therefore, based on the work of Nesterov and Todd, we discuss an implementation of a primal-dual interior-point method for solution of large-scale sparse conic quadratic optimization problems. The main features of the implementation are it is based on a homogeneous and self-dual model, handles the rotated quadratic cone directly, employs a Mehrotra type predictor-corrector

[*]Helsinki School of Economics and Business Administration, Department of Economics and Management Science, Runeberginkatu 14-16, Box 1210, FIN-00101 Helsinki, Finland, Email: e.d.andersen@mosek.com. Part of this research was done while the author had a TMR fellowship at TU Delft. This author has been supported by the Finnish Academy of Science

[†]TU Delft, Mekelweg 4, 2628 CD Delft, The Netherlands, Email: c.roos@its.tudelft.nl.

[‡]McMaster University, Department of Computing and Software, Hamilton, Ontario, Canada. L8S 4L7. Email: terlaky@cas.mcmaster.ca. Part of this research was done while the author was employed at TU Delft.

extension, and sparse linear algebra to improve the computational efficiency.

Computational results are also presented which documents that the implementation is capable of solving very large problems robustly and efficiently.

1 Introduction

Conic quadratic optimization is the problem of minimizing a linear objective function subject to the intersection of an affine set and the direct product of quadratic cones of the form

$$\left\{ x : \ x_1^2 \geq \sum_{j=2}^{n} x_j^2, \ x_1 \geq 0 \right\}. \tag{1}$$

The quadratic cone is also known as the second-order, the Lorentz, or the ice-cream cone.

Many optimization problems can be expressed in this form. Some examples are linear, convex quadratic, and convex quadratically constrained optimization. Other examples are the problem of minimizing a sum of norms and robust linear programming. Various applications of conic quadratic optimization are presented in [11, 17].

Over the last 15 years there has been extensive research into interior-point methods for linear optimization. One result of this research is the development of a primal-dual interior-point algorithm [16, 20] which is highly efficient both in theory and in practice [8, 18]. Therefore, several authors have studied how to generalize this algorithm to other problems. An important work in this direction is the paper of Nesterov and Todd [22] which shows that the primal-dual algorithm maintains its theoretical efficiency when the nonnegativity constraints are replaced by a convex cone as long as the cone is homogeneous and self-dual or in the terminology of Nesterov and Todd a self-scaled cone. It has subsequently been pointed out by Güler [15] that the only interesting cones having this property are direct products of R_+, the quadratic cone, and the cone of positive semi-definite matrices.

In the present work we will mainly focus on conic quadratic optimization and an algorithm for this class of problems.

Several authors have already studied algorithms for conic quadratic optimization. In particular Tsuchiya [27] and Monteiro and Tsuchiya [21] have studied the complexity of different variants of the primal-dual algorithm. Schmieta and Alizadeh [3] have shown that many of the polynomial algorithms developed for semi-definite optimization immediately can be translated to polynomial algorithms for conic quadratic optimization.

Andersen [9] and Alizadeh and Schmieta [2] discuss implementation of algorithms for conic quadratic optimization. Although they present good computational results then the implemented algorithms have an unknown complexity and cannot deal with primal or dual infeasible problems.

Sturm [26] reports that his code SeDuMi can perform conic quadratic and semi-definite optimization. Although the implementation is based on the work of Nesterov and Todd as described in [25] then only limited information is provided about how the code deals with the conic quadratic case.

The purpose of this paper is to present an implementation of a primal-dual interior-point algorithm for conic quadratic optimization which employs the best known algorithm (theoretically), which can handle large sparse problems, is robust, and handles primal or dual infeasible problems in a theoretically satisfactory way.

The outline of the paper is as follows. First we review the necessary duality theory for conic optimization and introduce the so-called homogeneous and self-dual model. Next we develop an algorithm based on the work of Nesterov and Todd for the solution of the homogeneous model. After presenting the algorithm we discuss efficient solution of the Newton equation system which has to be solved in every iteration of the algorithm. Indeed we show that the big Newton equation system can be reduced to solving a much smaller system of linear equations having a positive definite coefficient matrix. Finally, we discuss our implementation and present numerical results.

2 Conic optimization

2.1 Duality

In general a conic optimization problem can be expressed in the form

$$\begin{array}{llll} (P) & \text{minimize} & c^T x & \\ & \text{subject to} & Ax & = \; b, \\ & & x \in K & \end{array} \tag{2}$$

where K is assumed to be a pointed closed convex cone. Moreover, we assume that $A \in R^{m \times n}$ and all other quantities have conforming dimensions. For convenience and without loss of generality we will assume A is of full row rank. A primal solution x to (P) is said to be feasible if it satisfies all the constraints of (P). Problem (P) is feasible if it has at least one feasible solution. Otherwise the problem is infeasible. (P) is said to be strictly feasible if (P) has feasible solution such that $x \in \operatorname{int}(K)$, where $\operatorname{int}(K)$ denote the interior of K.

Let

$$K_* := \{s : \; s^T x \geq 0, \; \forall x \in K\} \tag{3}$$

be the dual cone, then the dual problem corresponding to (P) is given by

$$\begin{array}{llrcl} (D) & \text{maximize} & b^T y & & \\ & \text{subject to} & A^T y + s & = & c, \\ & & s \in K_*. & & \end{array} \tag{4}$$

A dual solution (y, s) is said to be feasible if it satisfies all the constraints of the dual problem. The dual problem (D) is feasible if it has at least one feasible solution. Moreover, (D) is strictly feasible if a dual solution (y, s) exists such that $s \in \operatorname{int}(K_*)$.

The following duality theorem is well-known:

Theorem 2.1 Weak duality: *Let x be a feasible solution to (P) and (y, s) be a feasible solution to (D), then*

$$c^T x - b^T y = x^T s \geq 0.$$

Strong duality: *If (P) is strictly feasible and its optimal objective value is bounded or (D) is strictly feasible and its optimal objective value is bounded, then (x, y, s) is an optimal solution if and only if*

$$c^T x - b^T y = x^T s = 0$$

and x is primal feasible and (y, s) is dual feasible.

Primal infeasibility: *If*

$$\exists (y, s) : \; s \in K_*, \; A^T y + s = 0, \; b^T y^* > 0, \tag{5}$$

then (P) is infeasible.

Dual infeasibility: *If*

$$\exists x : \; x \in K, \; Ax = 0, \; c^T x < 0, \tag{6}$$

then (D) *is infeasible.*

Proof: For a proof see for example [11]. □

The difference $c^T x - b^T y$ stands for the duality gap whereas $x^T s$ is called the complementarity gap. If $x \in K$ and $s \in K_*$, then x and s are said to be complementary if the corresponding complementarity gap is zero.

For a detailed discussion of duality theory in the conic case we refer the reader to [11].

2.2 A homogeneous model

The primal-dual algorithm for linear optimization suggested in [16, 20] and generalized by Nesterov and Todd [22] does not handle primal or dual infeasible problems very well. Indeed one assumption for the derivation of the algorithm is that both the primal and dual problem has strictly feasible solutions.

However, if a homogeneous model is employed then the problem about detecting infeasibility vanish. This model was first used by Goldman and Tucker [13] in their work for linear optimization. The idea of the homogeneous model is to embed the optimization problem into a slightly larger problem which always has a solution. Furthermore, an appropriate solution to the embedded problem either provides a certificate of infeasibility or a (scaled) optimal solution to the original problem. Therefore, instead of solving the original problem using an interior-point method, then the embedded problem is solved. Moreover, it has been shown that a primal-dual interior-point algorithm based on the homogeneous model works well in practice for the linear case, see [6, 28].

The Goldman-Tucker homogeneous model can be generalized as follows

$$\begin{array}{rcl} Ax - b\tau & = & 0, \\ A^T y + s - c\tau & = & 0, \\ -c^T x + b^T y - \kappa & = & 0, \\ (x;\tau) \in \bar{K}, (s;\kappa) \in \bar{K}_* & & \end{array} \tag{7}$$

to the case of conic optimization. Here we use the notation that

$$\bar{K} := K \times R_+ \quad \text{and} \quad \bar{K}_* := K_* \times R_+.$$

The homogeneous model (7) has been used either implicitly or explicitly in previous works. Some references are [12, 23, 25].

Subsequently we say a solution to (7) is complementary if the complementary gap

$$x^T s + \tau\kappa$$

is identical to zero.

Lemma 2.1 *Let $(x^*, \tau^*, y^*, s^*, \kappa^*)$ be any feasible solution to (7), then*

i)

$$(x^*)^T s^* + \tau^* \kappa^* = 0.$$

ii) If $\tau^* > 0$, then $(x^*, y^*, s^*)/\tau^*$ is a primal-dual optimal solution to (P).

iii) If $\kappa^* > 0$, then at least one of the strict inequalities

$$b^T y^* > 0 \tag{8}$$

and

$$c^T x^* < 0 \tag{9}$$

holds. If the first inequality holds, then (P) is infeasible. If the second inequality holds, then (D) is infeasible.

Proof: Statements i) and ii) are easy to verify. In the case $\kappa^* > 0$ one has

$$-c^T x^* + b^T y^* = \kappa^* > 0$$

which shows that at least one of the strict inequalities (8) and (9) holds. Now suppose (8) holds then we have that

$$\begin{array}{rcl} b^T y^* & > & 0, \\ A^T y^* + s^* & = & 0, \\ s^* \in K_* & & \end{array} \tag{10}$$

implying the primal problem is infeasible. Indeed, y^* is a Farkas type certificate of primal infeasibility. Finally, suppose that (9) holds, then

$$\begin{array}{rcl} c^T x^* & < & 0, \\ A x^* & = & 0, \\ x^* \in K & & \end{array} \tag{11}$$

and x^* is a certificate the dual infeasibility. □

This implies that any solution to the homogeneous model with

$$\tau^* + \kappa^* > 0 \tag{12}$$

is either a scaled optimal solution or a certificate of infeasibility. Therefore, an algorithm that solves (7) and computes such a solution is a proper solution algorithm for solving the conic optimization problems (P) and (D). If no such solution exists, then a tiny perturbation to the problem data exists such that the perturbed problem has a solution satisfying (12) [11]. Hence, the problem is ill-posed. In the case of linear optimization this is never the case. Indeed in this case a so-called strictly complementary solution satisfying (12) and $x^* + s^* > 0$ always exist. However, for example for a primal and dual feasible conic quadratic problem having non-zero duality gap, then (12) cannot be satisfied. See [11] for a concrete example.

3 Conic quadratic optimization

In the remaining part of this work we restrict our attention to cones which can be formulated as the product of R_+ and the quadratic cone. To be specific, we will work with the following three cones:

Definition 3.1 R_+**:**

$$R_+ := \{x \in R : \; x \geq 0\}. \tag{13}$$

Quadratic cone:

$$K^q := \{x \in R^n : \; x_1^2 \geq \|x_{2:n}\|^2 , \; x_1 \geq 0\}. \tag{14}$$

Rotated quadratic cone:

$$K^r := \{x \in R^n : \; 2 x_1 x_2 \geq \|x_{3:n}\|^2 , \; x_1, x_2 \geq 0\}. \tag{15}$$

These three cones are homogeneous and self-dual, see Definition A.1. Without loss of generality it can be assumed that

$$K = K^1 \times \ldots \times K^k.$$

i.e. the cone K is the direct product of several individual cones each one of the type (13), (14), or (15) respectively. Furthermore, let x be partitioned according to the cones i.e.

$$x = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^k \end{bmatrix} \quad \text{and} \quad x^i \in K^i \subseteq R^{n^i}.$$

Associated with each cone are two matrices

$$Q^i, T^i \in R^{n^i \times n^i}$$

which are defined in Definition 3.2.

Definition 3.2 *i.) If K^i is R_+, then*

$$T^i := 1 \quad \textit{and} \quad Q^i = 1. \tag{16}$$

ii.) If K^i is the quadratic cone, then

$$T^i := I_{n^i} \quad \textit{and} \quad Q^i := diag(1, -1, \ldots, -1). \tag{17}$$

iii.) If K^i is the rotated quadratic cone, then

$$T^i := \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & \cdots & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{18}$$

and

$$Q^i := \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -1 \end{bmatrix}. \tag{19}$$

It is an easy exercise to verify that each Q^i and T^i are orthogonal. Hence

$$Q^i Q^i = I \quad \text{and} \quad T^i T^i = I.$$

The definition of the Q matrices allows an alternative way of stating the quadratic cone because assume K^i is the quadratic cone then

$$K^i = \{x^i \in R^{n^i} : (x^i)^T Q^i x^i \geq 0, \ x_1^i \geq 0\}$$

and if K^i is a rotated quadratic cone, then

$$K^i = \{x^i \in R^{n^i} : (x^i)^T Q^i x^i \geq 0, \ x_1^i, x_2^i \geq 0\}.$$

If the ith cone is a rotated quadratic cone, then

$$x^i \in K^q \ \Leftrightarrow \ T^i x^i \in K^r.$$

which demonstrates that the rotated quadratic cone is identical to the quadratic cone under a linear transformation. This implies it is possible by introducing some additional variables and linear constraints to pose the rotated quadratic cone as a quadratic cone. However, for efficiency reason we will not do that but rather deal with the rotated cone directly. Nevertheless, from a theoretical point of view all the results for the rotated quadratic cone follow from the results for the quadratic cone.

For algorithmic purposes the complementarity conditions between the primal and dual solution are needed. Using the notation that if v is a vector, then capital V denotes a related "arrow head" matrix i.e.

$$V := \text{mat}\,(v) = \begin{bmatrix} v_1 & v_{2:n}^T \\ v_{2:n} & v_1 I \end{bmatrix} \quad \text{and} \quad v_{2:n} := \begin{bmatrix} v_2 \\ \vdots \\ v_n \end{bmatrix}.$$

The complementarity conditions can now be stated compactly as in Lemma 3.1.

Lemma 3.1 *Let $x, s \in K$ then x and s are complementary, i.e. $x^T s = 0$, if and only if*

$$X^i S^i e^i = S^i X^i e^i = 0, \quad i = 1, \ldots, k, \tag{20}$$

where $X^i := mat(T^i x^i)$, $S^i := mat(T^i s^i)$. $e^i \in R^{n^i}$ is the first unit vector of appropriate dimension.

Proof: See the Appendix. $\square$

Subsequently let X and S be two block diagonal matrices with X^i and S^i along the diagonal i.e.

$$X := \operatorname{diag}(X^1, \ldots, X^k) \quad \text{and} \quad S := \operatorname{diag}(S^1, \ldots, S^k).$$

Given $v \in \operatorname{int}(K)$, then it is easy to verify the following useful formula

$$\operatorname{mat}(v)^{-1} = V^{-1} = \frac{1}{v_1^2 - \|v_{2:n}\|^2} \begin{bmatrix} v_1 & -v_{2:n}^T \\ -v_{2:n} & \left(v_1 - \frac{\|v_{2:n}\|^2}{v_1}\right) I + \frac{v_{2:n} v_{2:n}^T}{v_1} \end{bmatrix}.$$

3.1 The central path

The guiding principle in primal-dual interior-point algorithms is to follow the so-called central path towards an optimal solution. The central path is a smooth curve connecting an initial point and a complementary solution. Formally, let an (initial) point $(x^{(0)}, \tau^{(0)}, y^{(0)}, s^{(0)}, \kappa^{(0)})$ be given such that

$$(x^{(0)}; \tau^{(0)}), (s^{(0)}; \kappa^{(0)}) \in \operatorname{int}(\bar{K})$$

then the set of nonlinear equations

$$\begin{array}{rcl} Ax - b\tau & = & \gamma(Ax^{(0)} - b\tau^{(0)}), \\ A^T y + s - c\tau & = & \gamma(A^T y^{(0)} + s^{(0)} - c\tau^{(0)}), \\ -c^T x + b^T y - \kappa & = & \gamma(-c^T x^{(0)} + b^T y^{(0)} - \kappa^{(0)}), \\ XSe & = & \gamma\mu^{(0)} e, \\ \tau\kappa & = & \gamma\mu^{(0)}, \end{array} \tag{21}$$

defines the central path parameterized by $\gamma \in [0, 1]$. Here $\mu^{(0)}$ is given by the expression

$$\mu^{(0)} := \frac{(x^{(0)})^T s^{(0)} + \tau^{(0)}\kappa^{(0)}}{k+1}.$$

and e by the expression

$$e := \begin{bmatrix} e^1 \\ \vdots \\ e^k \end{bmatrix}.$$

The first three blocks of equations in (21) are feasibility equations whereas the last two blocks of equations are the relaxed complementarity conditions.

In general it is not possible to compute a point on the central path exactly. However, using Newton's method a point in a neighborhood of the central path can be computed efficiently. Among the possible definitions of a neighborhood we will use the following definition

$$\mathcal{N}(\beta) := \left\{ (x,\tau,s,\kappa): \ (x;\tau),(s;\kappa) \in \bar{K}, \min \left(\begin{array}{c} \sqrt{(x^1)^T Q^1 x^1 (s^1)^T Q^1 s^1} \\ \vdots \\ \sqrt{(x^k)^T Q^k x^k (s^k)^T Q^k s^k} \\ \tau\kappa \end{array} \right) \geq \beta\mu \right\},$$

and

$$\mu := \frac{x^T s + \tau\kappa}{k+1}$$

where $\beta \in [0,1]$. Given this definition we can state Lemma 3.2.

Lemma 3.2 *i)* $\mathcal{N}(\beta) \subseteq \mathcal{N}(\beta')$ *where* $1 \geq \beta \geq \beta' \geq 0$.

ii) $(x;\tau),(s;\kappa) \in \mathcal{N}(1)$ *implies* $X^i S^i e^i = \mu e^i$ *and* $\tau\kappa = \mu$.

The interpretation of Lemma 3.2 is that the size of the neighborhood $\mathcal{N}(\beta)$ increases with the decrease in β. Moreover, the neighborhood $\mathcal{N}(1)$ conincides with the central path.

3.2 Scaling

For later use we need the definition of a scaling.

Definition 3.3 $W^i \in R^{n^i \times n^i}$ *is a scaling matrix if it satisfies the conditions*

$$\begin{array}{rcl} W^i & \succ & 0, \\ W^i Q^i W^i & = & Q^i, \end{array}$$

where $W^i \succ 0$ means W^i is symmetric and positive definite.

A scaled point $\bar{x}, \bar{s}$ is obtained by the transformation

$$\bar{x} := \Theta W x \quad \text{and} \quad \bar{s} := (\Theta W)^{-1} s,$$

where

$$W := \begin{bmatrix} W^1 & 0 & \cdots & 0 \\ 0 & W^2 & \vdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & W^k \end{bmatrix}$$

and

$$\Theta = \mathrm{diag}(\theta_1 1_{n^1}; \ldots; \theta_k 1_{n^k}).$$

1_{n^i} is the vector of all ones having the length n^i and $\theta \in R^k$.

Hence, W is a block diagonal matrix having the W^is along the diagonal and Θ is a diagonal matrix.

In Lemma 3.3 it is shown that scaling does not change anything. For example if the original point is in the interior of the cone K then the scaled point is in the interior too. Similarly, if the original point belongs to a certain neighborhood, then the scaled point belong to the same neighborhood.

Lemma 3.3 *i)* $(x^i)^T s^i = (\bar{x}^i)^T \bar{s}^i$.

ii) $\theta_i^2 (x^i)^T Q^i x^i = (\bar{x}^i)^T Q^i \bar{x}^i$.

iii) $\theta_i^{-2} (s^i)^T Q^i s^i = (\bar{s}^i)^T Q^i \bar{s}^i$.

iv) $x \in K \Leftrightarrow \bar{x} \in K$ *and* $x \in int(K) \Leftrightarrow \bar{x} \in int(K)$.

v) Given a $\beta \in (0,1)$ *then*

$$(x, \tau, s, \kappa) \in \mathcal{N}(\beta) \Rightarrow (\bar{x}, \tau, \bar{s}, \kappa) \in \mathcal{N}(\beta).$$

Proof: See the Appendix. □

4 The search direction

As mentioned previously the main algorithmic idea in a primal-dual interior-point algorithm is to trace the central path loosely. However, the central path is defined by the nonlinear equations (21) which cannot easily be solved, but an approximate solution can be computed using Newton's method. Indeed if one iteration of Newton's method is applied to (21) for a fixed γ, then a search direction $(d_x, d_\tau, d_y, d_s, d_\kappa)$ is obtained. This search direction is given as the solution to the linear equation system:

$$\begin{array}{rcl} Ad_x - bd_\tau & = & (\gamma - 1)(Ax^{(0)} - b\tau^{(0)}), \\ A^T d_y + d_s - cd_\tau & = & (\gamma - 1)(A^T y^{(0)} + s^{(0)} - c\tau^{(0)}), \\ -c^T d_x + b^T d_y - d_\kappa & = & (\gamma - 1)(-c^T x^{(0)} + b^T y^{(0)} - \kappa), \\ X^{(0)} T d_s + S^{(0)} T d_x & = & -X^{(0)} S^{(0)} e + \gamma \mu^{(0)} e, \\ \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau & = & -\tau^{(0)} \kappa^{(0)} + \gamma \mu^{(0)}. \end{array} \tag{22}$$

where

$$T := \begin{bmatrix} T^1 & 0 & \cdots & 0 \\ 0 & T^2 & \vdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & T^k \end{bmatrix}.$$

This direction is a slight generalization of the direction suggested in [1] to the homogeneous model. A new point is obtained by moving in the direction $(d_x, d_\tau, d_y, d_s, d_\kappa)$ as follows

$$\begin{bmatrix} x^{(1)} \\ \tau^{(1)} \\ y^{(1)} \\ s^{(1)} \\ \kappa^{(1)} \end{bmatrix} = \begin{bmatrix} x^{(0)} \\ \tau^{(0)} \\ y^{(0)} \\ s^{(0)} \\ \kappa^{(0)} \end{bmatrix} + \alpha \begin{bmatrix} d_x \\ d_\tau \\ d_y \\ d_s \\ d_\kappa \end{bmatrix} \tag{23}$$

for some step size $\alpha \in [0, 1]$. This is a promising idea because if the search direction is well-defined, then the new point will be closer to being feasible to the homogeneous model and complementary as shown in Lemma 4.1.

Lemma 4.1 *Given (22) and (23) then*

$$\begin{array}{rcl} Ax^{(1)} - b\tau^{(1)} & = & (1 - \alpha(1-\gamma))(Ax^{(0)} - b\tau^{(0)}), \\ A^T y^{(1)} + s^{(1)} - c\tau^{(1)} & = & (1 - \alpha(1-\gamma))(A^T y^{(0)} + s^{(0)} - c\tau^{(0)}), \\ -c^T x^{(1)} + b^T y^{(1)} - \kappa^{(1)} & = & (1 - \alpha(1-\gamma))(-c^T x^{(0)} + b^T y^{(0)} - \kappa^{(0)}), \\ d_x^T d_s^T + d_\tau d_\kappa & = & 0, \\ (x^{(1)})^T s^{(1)} + \tau^{(1)}\kappa^{(1)} & = & (1 - \alpha(1-\gamma))((x^{(0)})^T s^{(0)} + \tau^{(0)}\kappa^{(0)}). \end{array} \tag{24}$$

Proof: The first three equalities are trivial to prove, so we will only prove the last two equalities. First observe that

$$\begin{array}{rcl} A(\eta x^{(0)} + d_x) - b(\eta\tau^{(0)} + d_\tau) & = & 0, \\ A^T(\eta y^{(0)} + d_y) + (\eta s^{(0)} + d_s) - c(\eta\tau^{(0)} + d_\tau) & = & 0, \\ -c(\eta x^{(0)} + d_x) + b^T(\eta y^{(0)} + d_y) - (\eta\kappa^{(0)} + d_\kappa) & = & 0, \end{array}$$

where

$$\eta := 1 - \gamma. \tag{25}$$

This implies

$$\begin{array}{rcl} 0 & = & (\eta x^{(0)} + d_x)^T(\eta s^{(0)} + d_s) + (\eta\tau^{(0)} + d_\tau)(\eta\kappa^{(0)} + d_\kappa) \\ & = & \eta^2((x^{(0)})^T s^{(0)} + \tau^{(0)}\kappa^{(0)}) \\ & & +\eta((x^{(0)})^T d_s + (s^{(0)})^T d_x + \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau) \\ & & +d_x^T d_s + d_\tau d_\kappa. \end{array} \tag{26}$$

Moreover,

$$\begin{array}{rcl}(x^{(0)})^T d_s + (s^{(0)})^T d_x + \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau & = & e^T(X^{(0)} T d_s + S^{(0)} T d_x) + \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau \\ & = & e^T(-X^{(0)} S^{(0)} e + \gamma \mu^{(0)} e) - \tau^{(0)} \kappa^{(0)} + \gamma \mu^{(0)} \\ & = & (\gamma - 1)\mu^{(0)} k.\end{array}$$

These two facts combined gives

$$d_x^T d_s + d_\tau d_\kappa = 0$$

and

$$(x^{(1)})^T s^{(1)} + \tau^{(1)} \kappa^{(1)} = (1 - \alpha(1-\gamma))((x^{(0)})^T s^{(0)} + \tau^{(0)} \kappa^{(0)}).$$

□

Unfortunately the Newton search direction is only guaranteed to be well-defined in a narrow neighborhood around the central path [21]. However, one way to make sure that the search-direction is well-defined is to scale the problem appropriately before applying Newton's method and then scale the resulting search direction back to the original space. The resulting search direction belongs to the Monteiro-Zhang family of search directions and is defined by the linear equation system

$$\begin{array}{rcl} A d_x - b d_\tau & = & (\gamma - 1)(A x^{(0)} - b \tau^{(0)}), \\ A^T d_y + d_s - c d_\tau & = & (\gamma - 1)(A^T y^{(0)} + s^{(0)} - c \tau^{(0)}), \\ -c^T d_x + b^T d_y - d_\kappa & = & (\gamma - 1)(-c^T x^{(0)} + b^T y^{(0)} - \kappa), \\ \bar{X}^{(0)} T (\Theta W)^{-1} d_s + \bar{S}^{(0)} T \Theta W d_x & = & -\bar{X}^{(0)} \bar{S}^{(0)} e + \gamma \mu^{(0)} e, \\ \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau & = & -\tau^{(0)} \kappa^{(0)} + \gamma \mu^{(0)}. \end{array} \tag{27}$$

Given A is of full row rank and an appropriate choice of the scaling ΘW then it can be shown that the scaled Newton direction is uniquely defined. Moreover, all the properties stated in Lemma 4.1 are true for the scaled search direction as well. Finally, polynomial complexity can be proven, see for example Monteiro and Tuschiya [21]. Among the different possible scalings analysed in [21] the best results are obtained using NT scaling suggested in [22]. In the NT scaling ΘW is chosen such that

$$\bar{x} = \bar{s}, \tag{28}$$

holds which is equivalent to require that the scaled primal and dual points are identical. Note that the relation (28) implies

$$s = (\Theta^2 W^2) x. \tag{29}$$

In the case of NT scaling both Θ and W can be computed cheaply for each of our cones as demonstrated in Lemma 4.2.

Lemma 4.2 *Assume that $x^i, s^i \in int(K^i)$ then*

$$\theta_i^2 = \sqrt{\frac{(s^i)^T Q^i s^i}{(x^i)^T Q^i x^i}}. \tag{30}$$

Moreover, if K^i is

i) *the positive half-line R_+, then:*

$$W^i = \frac{1}{\theta_i}((X^i)^{-1} S^i)^{\frac{1}{2}}.$$

ii) *a quadratic cone, then:*

$$\begin{aligned} W^i &= \begin{bmatrix} w_1^i & (w_{2:n^i}^i)^T \\ w_{2:n^i}^i & I + \frac{w_{2:n^i}^i (w_{2:n^i}^i)^T}{1+w_1^i} \end{bmatrix} \\ &= -Q^i + \frac{(e_1^i+w^i)(e_1^i+w^i)^T}{1+(e_1^i)^T w^i} \end{aligned} \tag{31}$$

where

$$w^i = \frac{\theta_i^{-1} s^i + \theta_i Q^i x^i}{\sqrt{2}\sqrt{(x^i)^T s^i + \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i}}}. \tag{32}$$

Furthermore,

$$(W^i)^2 = -Q^i + 2w^i(w^i)^T. \tag{33}$$

iii) *a rotated quadratic cone, then:*

$$W^i = -Q^i + \frac{(T^i e_1^i + w^i)(T^i e_1^i + w^i)^T}{1 + (e_1^i)^T T^i w^i} \tag{34}$$

where w^i is given by (32). Furthermore,

$$(W^i)^2 = -Q^i + 2w^i(w^i)^T. \tag{35}$$

Proof: In case of the quadratic cone the Lemma is derived in [27], but we prefer to include a proof here for completeness. See the Appendix details. □

Lemma 4.3 *Let be W^i be given as in Lemma 4.2 then*

$$(\theta_i W^i)^{-2} = \theta_i^{-2} Q^i (W^i)^2 Q^i.$$

Proof: Using Definition 3.3 we have that $W^i Q^i W^i = Q^i$ and $Q^i Q^i = I$ which implies $(W^i)^{-1} = Q^i W^i Q^i$ and $(W^i)^{-2} = Q^i (W^i)^2 Q^i$. $\square$

One observation which can be made from Lemma 4.2 and Lemma 4.3 is that the scaling matrix W can be stored by using an n dimensional vector because only the vector w^i has to be stored for each cone. Furthermore, any multiplication with W or W^2 or their inverses can be carried out in $O(n)$ complexity. This is an important fact that should be exploited in an implementation.

4.1 Choice of the step size

After the search direction has been computed then a step size has to be chosen. It can be shown given the primal-dual algorithm is initiated with a solution sufficiently close to the central path and γ is sufficiently close to 1, then the unit step size ($\alpha = 1$) is always a suitable choice which also makes sure that the iterates stay in a close neighborhood of the central path.

However, in practice an aggressive choice of γ give rise to vastly improved performance. This makes it necessary to use a step size smaller than one, because otherwise the updated solution may move too far from the central path and even be infeasible with respect to the cone constraint.

In general the step size α should be chosen such that

$$(x^{(1)}, \tau^{(1)}, s^{(1)}, \kappa^{(1)}) \in \mathcal{N}(\beta) \tag{36}$$

where $\beta \in (0, 1)$ is a fixed constant. Next we will discuss how to compute the step size to satisfy this requirement.

First define

$$v_i^{x1} := (x^i)^T Q^i x^i, \quad v_i^{x2} := 2 d_{x^i}^T Q^i x^i, \quad \text{and} \quad v_i^{x3} := d_{x^i}^T Q^i d_{x^i}$$

and define v_i^{s1}, v_i^{s2}, and v_i^{s3} in a similar way. Next define

$$f_i^x(\alpha) := (x^i + \alpha d_{x^i})^T Q^i (x^i + \alpha d_{x^i}) = v_i^{x1} + \alpha v_i^{x2} + \alpha^2 v_i^{x3}, \tag{37}$$

and

$$f_i^s(\alpha) := (s^i + \alpha d_{s^i})^T Q^i (s^i + \alpha d_{s^i}) = v_i^{s1} + \alpha v_i^{s2} + \alpha^2 v_i^{s3}. \tag{38}$$

Note given the v vectors have been computed, then $f_i^x(\cdot)$ and $f_i^s(\cdot)$ can be evaluated in $O(k)$ complexity. Now define $\alpha^{\max}$ such that it is maximal and satisfies

$$\begin{array}{rcl} (e^i)^T(x^{(0)} + \alpha d_x) & \geq & 0, \quad \forall i, \alpha[0, \alpha^{\max}], \\ (e^i)^T(s^{(0)} + \alpha d_s) & \geq & 0, \quad \forall i, \alpha[0, \alpha^{\max}], \\ (e^i)^T Q^i(x^{(0)} + \alpha d_x) & \geq & 0, \quad \forall i, \alpha[0, \alpha^{\max}], \\ (e^i)^T Q^i(s^{(0)} + \alpha d_s) & \geq & 0, \quad \forall i, \alpha[0, \alpha^{\max}], \\ \tau + \alpha d_\tau & \geq & 0, \quad \forall \alpha[0, \alpha^{\max}], \\ \kappa + \alpha d_\kappa & \geq & 0, \quad \forall \alpha[0, \alpha^{\max}], \\ f_x(\alpha) & \geq & 0, \quad \forall \alpha[0, \alpha^{\max}], \\ f_s(\alpha) & \geq & 0, \quad \forall \alpha[0, \alpha^{\max}]. \end{array}$$

The purpose of the first four inequalities is to make sure that the appropriate elements of x and s stay positive. The choice of $\alpha^{\max}$ implies that for any $\alpha \in (0, \alpha^{\max})$ we have

$$(x^{(1)}; \tau^{(1)}), (s^{(1)}; \kappa^{(1)}) \in \operatorname{int}(\bar{K}).$$

Next a decreasing sequence of α_l's for $l = 1, 2, \ldots$ in the interval $(0, \alpha^{\max})$ is chosen and the largest element in the sequence which satisfies

$$\begin{array}{rcl} \sqrt{f_x(\alpha_l) f_s(\alpha_l)} & \geq & \beta(1 - \alpha_l(1-\gamma))\mu^{(0)}, \\ (\tau^{(0)} + \alpha_l d_\tau)(\kappa^{(0)} + \alpha_l d_\kappa) & \geq & \beta(1 - \alpha_l(1-\gamma))\mu^{(0)}, \end{array} \tag{39}$$

is chosen as the step size. Enforcing the condition (39) is equivalent to enforce the condition (36).

4.2 Adapting Mehrotra's predictor-corrector method

Several important issues have not been addressed so far. In particular nothing has been stated about the choice of γ. In theoretical work on primal-dual interior-point algorithms γ is usually chosen as a constant close to one but in practice this leads to slow convergence. Therefore, in the linear case Mehrotra [19] suggested a heuristic which chooses γ dynamically depending on how much progress that can be made in the pure Newton (affine scaling) direction. Furthermore, Mehrotra suggests using a second-order correction of the search direction which increases the efficiency of the algorithm significantly in practice [18].

In this section we discuss how these two ideas proposed by Mehrotra can be adapted to the primal-dual method based on the Monteiro-Zhang family of search directions.

Mehrotra's predictor-corrector method utilizes the observation that

$$\mathrm{mat}\,(Tx) + \mathrm{mat}\,(Td_x) = \mathrm{mat}\,(T(x+d_x))$$

which implies

$$\begin{array}{rcl}(X+D_x)(S+D_s)e & = & \mathrm{mat}\,(T(x+d_x))\,\mathrm{mat}\,(T(s+d_s))\,e \\ & = & XSe + SD_xe + XD_se + D_xD_se,\end{array}$$

where

$$D_x := \mathrm{mat}\,(Td_x) \quad \text{and} \quad D_s := \mathrm{mat}\,(Td_s)\,.$$

When Newton's method is applied to the perturbed complementarity conditions

$$XS = \gamma\mu^{(0)}e$$

then the quadratic term

$$D_xD_se \tag{40}$$

is neglected and the search direction is obtained by solving the resulting system of linear equations. Instead of neglecting the quadratic term, then Mehrotra suggests estimate it using the pure Newton direction. Indeed, Mehrotra suggests to compute the primal-dual affine scaling direction

$$(d_x^n, d_\tau^n, d_y^n, d_s^n, d_\kappa^n)$$

first which is the unique solution of (27) for $\gamma = 0$. Next this direction is used to estimate the quadratic term as follows

$$D_xD_se \approx D_x^nD_s^ne \quad \text{and} \quad d_\tau d_\kappa \approx d_\tau^n d_\kappa^n.$$

In the framework of the Monteiro-Zhang family of search directions this implies that the linearized complementarity conditions in (27) are replaced by

$$\begin{array}{rcl}\bar{X}^{(0)}T(\Theta W^{-1}d_s + \bar{S}^{(0)}T\Theta W d_x & = & -\bar{X}^{(0)}\bar{S}^{(0)}e + \gamma\mu^{(0)}e - \bar{D}_x^n\bar{D}_s^ne, \\ \tau^{(0)}d_\kappa + \kappa^{(0)}d_\tau & = & -\tau^{(0)}\kappa^{(0)} + \gamma\mu^{(0)} - d_\tau^n d_\kappa^n\end{array}$$

where

$$\bar{D}_x^n := \mathrm{mat}\,(T\Theta W d_x^n) \quad \text{and} \quad \bar{D}_s^n := \mathrm{mat}\left(T(\Theta W)^{-1}d_s^n\right).$$

Note that even though the corrector term is included in the right-hand side then it can be proved that the final search direction satisfies all the properties stated in Lemma 4.1.

Mehrotra suggests another use of the pure Newton direction because he suggests to use it for a dynamic choice of γ based on how much progress that can be made in the affine scaling direction. Now let $\alpha_n^{\max}$ be the maximum step size to the boundary which can be taken along the pure Newton direction. According to Lemma 4.1 this implies that the residuals and the complementarity gap are reduced by a factor of

$$1 - \alpha_n^{\max}.$$

Then it seems reasonable to choose γ small if $\alpha_n^{\max}$ is large. The heuristic

$$\gamma = \min(\delta, (1 - \alpha_n^{\max})^2)(1 - \alpha_n^{\max})$$

achieve this, where $\delta \in [0, 1]$ is a fixed constant.

4.3 Adapting Gondzio's centrality correctors

In Mehrotra's predict-corrector method the search direction is only corrected once. An obvious idea is to repeat the corrections several times to obtain a high-order search directions. However, most of the computational experiments with this idea has not been successful. More recently Gondzio [14] has suggested another modification to the primal-dual algorithm for linear optimization which employs so-called centrality correctors. The main idea underlying this approach is to compute corrections to the search direction in such a way that the step size increases and hence faster convergence of the algorithm is achieved.

The idea of centrality correctors can successfully be adapted to the case when the earlier presented homogeneous primal-dual algorithm is applied to a linear optimization problem[6]. Therefore, in the present section we will discuss how to adapt the centrality corrector idea to the case when the optimization problem contains quadratic cones.

As mentioned previously then the iterates generated by our algorithm should stay in a close neighborhood of the central path or, ideally,

$$(x^{(1)}, \tau^{(1)}, s^{(1)}, \kappa^{(1)}) \in \mathcal{N}(1).$$

This implies that we are targeting

$$\sqrt{(x^{i(1)})^T Q^i x^{i(1)} (s^{i(1)})^T Q^i s^{i(1)}} = \tau^{(1)}\kappa^{(1)} = \mu^{(1)}.$$

However, this is in general a too ambitious target to reach and many iterations of Newton's method may be required to compute a good approximation of such a point. However, it might be possible to reach the less ambitious target

$$\mu^l \leq \begin{bmatrix} \sqrt{(x_1^{(1)})^T Q^1 x_1^{(1)} (s_1^{(1)})^T Q^1 s_1^{(1)}} \\ \vdots \\ \sqrt{(x_k^{(1)})^T Q^k x_k^{(1)} (s_k^{(1)})^T Q^k s_k^{(1)}} \\ \tau^{(1)}\kappa^{(1)} \end{bmatrix} \leq \mu^u \tag{41}$$

where μ^l and μ^u are suitably chosen constants. We will use the perhaps natural choice

$$\mu^l = \lambda\gamma\mu^{(0)} \quad \text{and} \quad \mu^u = \lambda^{-1}\gamma\mu^{(0)}$$

for some $\lambda \in (0, 1)$.

Now assume a γ and a search direction $(d_x, d_\tau, d_y, d_s, d_\kappa)$ have been computed as discussed in the previous sections and if for example

$$\sqrt{f_i^x(\hat{\alpha}) f_i^s(\hat{\alpha})} < \mu^l \tag{42}$$

for a reasonably chosen $\hat{\alpha}$, then we would like to compute a modification of the search direction such that the left-hand side of (42) is increased when the corrected search direction is employed.

This aim can be achieved as follows. First define

$$f^i = \begin{cases} \mu^l e^i - (\bar{X}^{(0)} + \hat{\alpha}\bar{D}_x)(\bar{S}^{(0)} + \hat{\alpha}\bar{D}_s)e^i, & f_i^x(\hat{\alpha}) f_i^s(\hat{\alpha}) \leq (\mu^l)^2, \\ \mu^u e^i - (\bar{X}^{(0)} + \hat{\alpha}\bar{D}_x)(\bar{S}^{(0)} + \hat{\alpha}\bar{D}_s)e^i, & f_i^x(\hat{\alpha}) f_i^s(\hat{\alpha}) \geq (\mu^u)^2, \\ 0e^i, & \text{otherwise} \end{cases} \tag{43}$$

and

$$f_{\tau\kappa} = \begin{cases} \mu^l - (\tau^{(0)} + \hat{\alpha}d_\tau)(\kappa^{(0)} + \hat{\alpha}d_\kappa), & (\tau^{(0)} + \hat{\alpha}d_\tau)(\kappa^{(0)} + \hat{\alpha}d_\kappa) \leq \mu^l, \\ \mu^u - (\tau^{(0)} + \hat{\alpha}d_\tau)(\kappa^{(0)} + \hat{\alpha}d_\kappa), & (\tau^{(0)} + \hat{\alpha}d_\tau)(\kappa^{(0)} + \hat{\alpha}d_\kappa) \geq \mu^u, \\ 0, & \text{otherwise.} \end{cases} \tag{44}$$

Moreover, let

$$f := \begin{bmatrix} f^1 \\ \vdots \\ f^k \end{bmatrix}$$

and

$$\mu^c := \gamma\mu^{(0)} - \frac{e^T f + f_{\tau\kappa}}{k+1}, \tag{45}$$

then we will define a corrected search direction by

$$\begin{array}{rcl} Ad_x - bd_\tau & = & (\gamma-1)(Ax^{(0)} - b\tau^{(0)}), \\ A^T d_y + d_s - cd_\tau & = & (\gamma-1)(A^T y^{(0)} + s^{(0)} - c\tau^{(0)}), \\ -c^T d_x + b^T d_y - d_\kappa & = & (\gamma-1)(-c^T x^{(0)} + b^T y^{(0)} - \kappa), \\ \bar{X}^{(0)} T(\Theta W)^{-1} d_s + \bar{S}^{(0)} T\Theta W d_x & = & -\bar{X}^{(0)}\bar{S}^{(0)} e + \mu^c e - \bar{D}^n_x \bar{D}^n_s + f, \\ \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau & = & -\tau^{(0)}\kappa^{(0)} + \mu^c - d^n_\tau d^n_\kappa + f_{\tau\kappa}. \end{array} \tag{46}$$

Note compared to the original search direction then only the right-hand side of the linearized complementarity conditions have been modified. Next let η be given by (25) then due to

$$\begin{array}{rcl} A(\eta x^{(0)} + d_x) - b(\eta\tau^{(0)} + d_\tau) & = & 0, \\ A^T(\eta y^{(0)} + d_y) + (\eta s^{(0)} + d_s) - c(\eta\tau^{(0)} + d_\tau) & = & 0, \\ -c^T(\eta x^{(0)} + d_x) + b^T(\eta y^{(0)} + d_y) - (\eta\kappa^{(0)} + d_\kappa) & = & 0, \end{array}$$

holds the orthogonality of the search direction holds as well i.e. (26) holds. Furthermore, we have that

$$\begin{array}{rl} & (x^{(0)})^T d_s + (s^{(0)})^T d_x + \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau \\ = & e^T(\bar{X}^{(0)} T(\Theta W)^{-1} d_s + \bar{S}^{(0)} T\Theta W d_x) + \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau \\ = & e^T(-\bar{X}^{(0)}\bar{S}^{(0)} e + \mu^c e - \bar{D}^n_x \bar{D}^n_s + f - \tau^{(0)}\kappa^{(0)} + \mu^c - d^n_\tau d^n_\kappa + f_{\tau\kappa}) \\ = & -(x^{(0)})^T s^{(0)} - \tau^{(0)}\kappa^{(0)} + \mu^c(1 + e^T e) - e^T \bar{D}^n_x \bar{D}^n_s e - d^n_\tau d^n_\kappa + e^T f + f_{\tau\kappa} \\ = & (\gamma-1)((x^{(0)})^T s^{(0)} + \tau^{(0)}\kappa^{(0)}) \end{array}$$

because of the fact

$$e^T \bar{D}^n_x \bar{D}^n_s e + d^n_\tau d^n_\kappa = 0$$

and the definition of t and $t_{\tau\kappa}$. The combination of these facts leads to the conclusion

$$d_x^T d_s + d_\tau d_\kappa = 0.$$

Hence, the search direction defined by (46) satisfies all the properties of Lemma 4.1.

After the corrected search direction defined by (46) has been computed then the maximal step size $\alpha^{\max}$ is recomputed. However, there is nothing which guarantees that the new maximal step size is larger than the step size corresponding to original search direction. If this is not the case, then the corrected search direction is discarded and the original direction is employed.

Clearly, this process of computing corrected directions can be repeated several times, where the advantage of computing several corrections is that the number of iterations (hopefully) is further decreased. However, the computation of the corrections is not free so there is a trade-off between the time it takes to compute an additional corrected search direction and the expected reduction in the number of iterations. Therefore, the maximum number of corrections computed is determined using a strategy similar to that of Gondzio [14]. Moreover, an additional correction is only computed if the previous correction increases the step size by more than 20%.

5 Computing the search direction

The computationally most expensive part of a primal-dual algorithm is the computation of the search direction because this involves the solution of a potentially very large system of linear equations. Indeed, in each iteration of the primal-dual algorithm, a system of linear equations of the form

$$\begin{array}{rcl} Ad_x - bd_\tau & = & r^1, \\ A^T d_y + d_s - cd_\tau & = & r^2, \\ -c^T d_x + b^T d_y - d_\kappa & = & r^3, \\ \bar{X}^{(0)} T(\Theta W)^{-1} d_s + \bar{S}^{(0)} T \Theta W d_x & = & r^4, \\ \tau^{(0)} d_\kappa + \kappa^{(0)} d_\tau & = & r^5 \end{array} \tag{47}$$

must be solved for several different right-hand sides r. An important fact is that for most large-scale problems appearing in practice, then the matrix A is sparse. This sparsity can and should be exploited to improve the computational efficiency.

Before proceeding with the details of the computation of the search direction then recall that any matrix-vector product involving the matrices $\bar{X}^{(0)}$, $\bar{S}^{(0)}$, T, W, Θ, W^2, and Θ^2, or their inverses, can be carried out in $O(n)$

complexity. Hence, these operations are computationally cheap operations and will not be considered further.

The system (47) can immediately be reduced by eliminating d_s and d_κ from the system using

$$\begin{array}{rcl} d_s &=& \Theta W T(\bar{X}^{(0)})^{-1}(r^4 - \bar{S}^{(0)} T \Theta W d_x), \\ d_\kappa &=& (\tau^{(0)})^{-1}(r^5 - \kappa^0 d_\tau). \end{array} \tag{48}$$

Next let (g^1, g^2) and (h^1, h^2) be defined as the solutions to

$$\begin{bmatrix} -(\Theta W)^2 & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} g^1 \\ g^2 \end{bmatrix} = \begin{bmatrix} c \\ b \end{bmatrix} \tag{49}$$

and

$$\begin{bmatrix} -(\Theta W)^2 & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} h^1 \\ h^2 \end{bmatrix} = \begin{bmatrix} r^2 - \Theta W T(\bar{X}^{(0)})^{-1} r^4 \\ r^1 \end{bmatrix}. \tag{50}$$

Given A is of full rank, then these two systems have a unique solution. Moreover, we have that

$$d_\tau = \frac{r^3 - c^T h^1 + b^T h^2}{(\tau^{(0)})^{-1}\kappa^{(0)} + c^T g^1 - b^T g^2}$$

and

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} g^1 \\ g^2 \end{bmatrix} + \begin{bmatrix} h^1 \\ h^2 \end{bmatrix} d_\tau.$$

After d_τ and d_x have been computed, then d_s and d_κ can be computed using relation (48). Therefore, given (49) and (50) can be solved efficiently, then the search direction can be computed efficiently as well. Now the solution to (49) and (50) is given by

$$\begin{array}{rcl} g^2 &=& (A(\Theta W)^{-2} A^T)^{-1}(b + A(\Theta W)^2 c), \\ g^1 &=& -(\Theta W)^{-2}(c - A^T g^2), \end{array}$$

and

$$\begin{array}{rcl} h^2 &=& (A(\Theta W)^{-2} A^T)^{-1}(r^1 + A(\Theta W)^2 (r^2 - (\Theta W T \bar{X}^{(0)})^{-1} r^4)), \\ h^1 &=& -(\Theta W)^{-2}(r^2 - \Theta W (\bar{X}^{(0)})^{-1} r^4 - A^T h^2), \end{array}$$

respectively. Hence, we have reduced the computation of the search direction to computing

$$(A(\Theta W)^{-2} A^T)^{-1}$$

or equivalently to solve a linear equation system of the form

$$Mh = f$$

where

$$M := A(\Theta W)^{-2} A^T.$$

Recall W is a block diagonal matrix having the W^is along the diagonal. Moreover, Θ is a positive definite diagonal matrix. This implies $(\Theta W)^2$ is a positive definite block diagonal matrix and hence M is symmetric and positive definite. Therefore, M has a Cholesky factorization i.e.

$$M = LL^T$$

where L is a lower triangular matrix. It is well-known that if the matrix M is sparse, then the Cholesky factorization can usually be computed efficiently in practice. This leads to the important questions whether M can be computed efficiently and whether M is likely to be sparse.

First observe that

$$M = A(\Theta W)^{-2} A^T = \sum_{i=1}^{k} \theta_i^{-2} A^i (W^i)^{-2} (A^i)^T,$$

where A^i is the columns of A corresponding to the variables in x^i. In the case the ith cone is R_+ then W^i is a scalar and A^i is a column vector. This implies the term

$$A^i (W^i)^{-2} (A^i)^T$$

can easily be computed and is sparse if A^i is sparse. In the case the ith cone is a quadratic or a rotated quadratic cone then

$$\begin{array}{rcl} A^i (W^i)^{-2} (A^i)^T & = & A^i Q^i (-Q^i + 2 w^i (w^i)^T) Q^i (A^i)^T \\ & = & -A^i Q^i (A^i)^T + 2 (A^i Q^i w^i)(A^i Q^i w^i)^T, \end{array} \tag{51}$$

which is a sum of two terms. The term

$$(A^i Q^i w^i)(A^i Q^i w^i)^T$$

is sparse if the vector

$$A^i Q^i w^i \tag{52}$$

is sparse and A^i contains no dense columns. Note if n^i is large, then it cannot be expected that (52) is sparse, because the sparsity pattern of (52) is identical to the union of the sparsity patterns of all the columns in A^i. The term

$$A^i Q^i (A^i)^T \tag{53}$$

also tends to be sparse. Indeed in the case the ith cone is a quadratic cone then the sparsity pattern of (53) is identical to the sparsity pattern of $A^i(A^i)^T$ which is likely to be sparse given A^i contains no dense columns. In the case of the rotated quadratic cone we have that

$$A^i Q^i (A^i)^T = A^i_{:1}(A^i_{:2})^T + A^i_{:2}(A^i_{:1})^T - A^i_{:(3:n^i)}(A^i_{:(3:n^i)})^T.$$

This term is likely to be sparse except if the union of the sparsity patterns in $A^i_{:1}$ and $A^i_{:2}$ are dense or A^i contain dense columns. ($A^i_{:j}$ denotes the jth column of A^i and $A^i_{:(j:k)}$ denotes the columns j to k of A^i.)

In summary if all the vectors w^i are of low dimmension and the columns of A^i are sparse, then M can be expected to be sparse which implies that the Newton equation system can be solved very efficiently using the Cholesky factorization.

In the computational results reported in Section 8 we use this approach. Details about how the Cholesky decomposition is computed can be seen in [5, 6].

5.1 Exploiting structure in the constraint matrix

In practice most optimization problems have some structure in the constraint matrix which can be exploited to speed up the computations. In our implementation we exploit the following two types of constraint structures:

Upper bound constraints: If a variable x_j has both a lower and an upper bound, then an additional constraint of the form

$$x_j + x_k = u$$

must be introduced where x_k is a slack variable and therefore occurs in only one constraint.

Singleton constraints: For reasons that becomes clear in Section 7.1, constraints of the form

$$x_j = b$$

frequently arises where x_j only occurs in one constraint. Moreover, we will assume that x_j does not belong to a linear cone because such variable can simply be substituted out of the problem.

Exploiting the upper bound constraints is trivial and similar to the pure linear optimization case as discussed in for example [7]. Therefore, we will not discuss this case further and the subsequent discussion is limited to the singleton constraints case only.

After a suitable reordering of the variables and the constraints, we may assume A has the form

$$A = \begin{bmatrix} 0 & A_{12} \\ I & 0 \end{bmatrix}$$

where

$$\begin{bmatrix} I & 0 \end{bmatrix}$$

corresponds to the set of singleton constraints. This observation can be exploited when solving the systems (49) and (50). Subsequently we will demonstrate how to do this for the system (49).

First assume that the vector g and the right-hand side of the system has been partitioned similar to A and according to the partition of

$$H := \begin{bmatrix} -H_{11} & -H_{12} \\ -H_{21} & -H_{22} \end{bmatrix} = -(\Theta W)^{-2}.$$

This implies the system (49) may be written as

$$\begin{bmatrix} -H_{11} & -H_{12} & 0 & I \\ -H_{21} & -H_{22} & A_{12}^T & 0^T \\ 0 & A_{12} & 0 & 0 \\ I & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} g_1^1 \\ g_2^1 \\ g_1^2 \\ g_2^2 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ b_1 \\ b_2 \end{bmatrix}.$$

This large system can be reduced to the two small systems

$$\begin{bmatrix} -H_{22} & A_{12}^T \\ A_{12} & 0 \end{bmatrix} \begin{bmatrix} g_1^2 \\ g_2^1 \end{bmatrix} = \begin{bmatrix} c_2 \\ b_1 \end{bmatrix} + \begin{bmatrix} H_{21} b_2 \\ 0 \end{bmatrix} \tag{54}$$

and

$$\begin{bmatrix} -H_{11} & I^T \\ I & 0 \end{bmatrix} \begin{bmatrix} g_1^1 \\ g_2^2 \end{bmatrix} = \left(\begin{bmatrix} c_1 \\ b_2 \end{bmatrix} - \begin{bmatrix} -H_{12} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} g_1^2 \\ g_2^1 \end{bmatrix} \right) \tag{55}$$

that has to be solved in the order as stated. First observe that the second system (55) is trivial to solve whereas system (54) is easily solved if the inverses or appropriate factorizations of the matrices

$$H_{22}$$

and

$$A_{21}H_{22}^{-1}A_{21}^T \tag{56}$$

are known. Next observe that A_{21} is of full row rank because A is of full row rank. Finally, due to H is positive definite, then H_{22} is positive definite which implies that matrix (56) is positive definite.

Since matrix H_{22} is identical to H, except some rows and columns have been removed, then H_{22} is also a block diagonal matrix where each block originate from a block in H. Subsequently we will show that the inverse of each block in H_{22} can be computed efficiently.

In the discussion we will assume that H_{22} consists of one block only. It should be obvious how to extend the discussion to the case of multiple blocks. Any block in H can be written in the form

$$-Q + 2ww^T,$$

where we have dropped the cone subscript i for convenience. Here Q is either of the form (17) or (19) and w is a vector. Next we will partition the block to obtain

$$-\begin{bmatrix} Q_{11} & Q_{21} \\ Q_{12} & Q_{22} \end{bmatrix} + 2\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}^T.$$

After dropping the appropriate rows and columns we assume we are left with the H_{22} block

$$-Q_{11} + 2w_1w_1^T$$

which we have to compute an inverse of. First assume Q_{11} is nonsingular then by the Sherman-Morrison-Woodbury formula[1] we have that

$$(-Q_{11} + 2w_1w_1^T)^{-1} = -Q_{11}^{-1} - 2\frac{Q_{11}^{-1}w_1w_1^TQ_{11}^{-1}}{1 - 2w_1^TQ_{11}^{-1}w_1}$$

which is the required explicit representation for the inverse of the block.

[1]If the matrices B and matrix $B + vv^T$ are nonsingular, then $1 + v^TB^{-1}v$ is nonzero and $(B + vv^T)^{-1} = B^{-1} - \frac{B^{-1}vv^TB^{-1}}{1+v^TB^{-1}v}$.

In most cases Q_{11} is a nonsingular matrix, because it is only singular if the block corresponds to a rotated quadratic cone and either x_1 or x_2 but not both variables are fixed. This implies that in the case Q_{11} is singular then it can be assumed that Q_{11} has the form

$$Q_{11} = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}.$$

Now let

$$w_1 = \begin{bmatrix} \bar{w}_1 \\ \bar{w}_2 \end{bmatrix}$$

where $\bar{w}_1$ is a scalar and it can be verified that $\bar{w}_1 > 0$. It is now easy to verify that

$$-Q_{11} + 2w_1 w_1^T = FF^T$$

where

$$F := \begin{bmatrix} \sqrt{2}\bar{w}_1 & 0 \\ \sqrt{2}\bar{w}_2 & I \end{bmatrix} \quad \text{and} \quad F^{-1} = \begin{bmatrix} \frac{1}{\sqrt{2}\bar{w}_1} & 0 \\ -\frac{\bar{w}_2}{\bar{w}_1} & I \end{bmatrix}.$$

Hence,

$$\begin{aligned}
(-Q_{11} + 2w_1 w_1^T)^{-1} &= (FF^T)^{-1} \\
&= \begin{bmatrix} \frac{1+2\|\bar{w}_2\|^2}{2\bar{w}_1^2} & -\frac{\bar{w}_2^T}{\bar{w}_1} \\ -\frac{\bar{w}_2}{\bar{w}_1} & I \end{bmatrix} \\
&= \begin{bmatrix} \frac{1+2\|\bar{w}_2\|^2}{2\bar{w}_1^2} & 0 \\ 0 & I \end{bmatrix} - \frac{1}{\bar{w}_1}\left(\begin{bmatrix} 0 \\ \bar{w}_2 \end{bmatrix} e_1^T + e_1 \begin{bmatrix} 0 \\ \bar{w}_2 \end{bmatrix}^T \right),
\end{aligned}$$

which is the explicit representation of the inverse of the H_{22} block we were looking for. In summary instead of computing a factorization of $A(\Theta W)^{-2}A^T$, then it is sufficient to compute a factorization of the potentially much smaller matrix (56) plus additional cheap linear algebra.

6 Starting and stopping

6.1 Starting point

In our implementation we use the simple starting point:

$$x^{i(0)} = s^{i(0)} = T^i e_1^i.$$

Moreover, let $y^{(0)} = 0$ and $\tau^{(0)} = \kappa^{(0)} = 1$. This choice of starting point implies that

$$(x^{(0)}, \tau^{(0)}, s^{(0)}, \kappa^{(0)}) \in \mathcal{N}(1).$$

In the special case of linear optimization, the algorithm presented here is equivalent to the algorithm studied in Andersen and Andersen [6]. However, they employ a different starting point, than the one suggested above, which improves the practical efficiency for the linear case. Therefore, it might be possible to develop another starting point which works better for most problems occurring in practice. However, this is a topic left for future research.

6.2 Stopping criteria

An important issue is when to terminate the interior-point algorithm. Obviously the algorithm cannot be terminated before a feasible solution to the homogeneous model has been obtained. Therefore, to measure the infeasibility the following measures

$$\begin{array}{rcl} \rho_P^{(k)} & := & \frac{\left\| Ax^{(k)} - b\tau^{(k)} \right\|}{\max(1, \left\| Ax^{(0)} - b\tau^{(0)} \right\|)}, \\ \rho_D^{(k)} & := & \frac{\left\| A^T y^{(k)} + s^{(k)} - c\tau^{(k)} \right\|}{\max(1, \left\| A^T y^{(0)} + s^{(0)} - c\tau^{(0)} \right\|)}, \\ \rho_G^{(k)} & := & \frac{|-c^T x^{(k)} + b^T y^{(k)} - \kappa^{(k)}|}{\max(1, |-c^T x^{(0)} + b^T y^{(0)} - \kappa^{(0)}|)} \end{array}$$

are employed which essentially measure the relative reduction in the primal, dual, and gap infeasibility respectively. Also define

$$\rho_A^{(k)} := \frac{|c^T x^{(k)} - b^T y^{(k)}|}{\tau^{(k)} + |b^T y^{(k)}|} = \frac{|c^T x^{(k)}/\tau^{(k)} - b^T y^{(k)}/\tau^{(k)}|}{1 + |b^T y^{(k)}/\tau^{(k)}|} \tag{57}$$

which measures the number of significant digits in the objective value. The kth iterate is considered nearly feasible and optimal if

$$\rho_P^{(k)} \le \bar{\rho}_P, \quad \rho_D^{(k)} \le \bar{\rho}_D, \quad \text{and} \quad \rho_A^k \le \bar{\rho}_A,$$

where $\bar{\rho}_P, \bar{\rho}_D, \bar{\rho}_A \in (0, 1]$ are (small) user specified constants. In this case the solution

$$(x^*, y^*, s^*) = (x^{(k)}, y^{(k)}, s^{(k)})/\tau^{(k)}$$

is reported to be the optimal solution to (P).

The algorithm is also terminated if

$$\rho_P^{(k)} \leq \bar{\rho}_P, \quad \rho_D^{(k)} \leq \bar{\rho}_D, \quad \rho_G^{(k)} \leq \bar{\rho}_G, \quad \text{and} \quad \tau^{(k)} \leq \bar{\rho}_I \max(1, \kappa^{(k)}),$$

where $\bar{\rho}_I \in (0,1)$ is a small user specified constant. In this case a feasible solution to the homogeneous model with a small τ has been computed. Therefore, it is concluded that the problem is primal or dual infeasible. If $b^T y^{(k)} > 0$, then the primal problem is concluded to be infeasible and if $c^T x^{(k)} < 0$, then the dual problem is concluded to be infeasible. Moreover, the algorithm is terminated if

$$\mu^{(k)} \leq \bar{\rho}_\mu \mu^{(0)} \quad \text{and} \quad \tau^{(k)} \leq \bar{\rho}_I \min(1, \kappa^{(k)})$$

and the problem is reported to be ill-posed. $\bar{\rho}_A, \bar{\rho}_G, \bar{\rho}_\mu, \bar{\rho}_I \in (0,1]$ are all user specified constants.

7 Implementation

The algorithm for solving conic quadratic optimization problems have now been presented in details and we will therefore turn the attention to a few implementational issues.

7.1 Input format

In practice an optimization problem is usually specified by using an MPS file or a modeling language such as AIIMS, AMPL, GAMS, or MPL. However, none of these formats allow the user to specify that a part of x belongs to a quadratic cone (or a semi-definite cone). Indeed they can only handle constraints of the type[2]

$$g(x) \leq 0.$$

However, the two quadratic cones the proposed algorithm can handle is given by

$$\{x \in R^n : \; x_1^2 \geq \|x_{2:n}\|^2, \; x_1 \geq 0\}$$

and

$$\{x \in R^n : \; 2x_1 x_2 \geq \|x_{3:n}\|^2, \; x_1, x_2 \geq 0\}$$

[2]Of course these formats can also handle equalities, but that does not help.

which are quadratic constraints of the form

$$g(x) = \frac{1}{2}x^T Qx + a^T x + b \leq 0. \tag{58}$$

plus some additional bounds on the variables. Hence, it is possible to specify any conic quadratic optimization problem using quadratic inequalities. Therefore, any input format which allows description of linear and quadratic constraints can be used as an input format. Such an input format will of course also allow the specification of nonconvex quadratic problems and therefore it is necessary to require that Q has a particular form. In our implementation we require that Q has the form

$$\begin{bmatrix} q_{11} & 0 & \cdots & 0 \\ 0 & q_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & q_{nn} \end{bmatrix} \tag{59}$$

or

$$\begin{bmatrix} q_{11} & 0 & \dots & \cdots & \cdots & \cdots & 0 \\ 0 & q_{22} & \dots & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & 0 & q_{ij} & \cdots & 0 \\ \vdots & \vdots & \vdots & q_{ji} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & q_{nn} \end{bmatrix}. \tag{60}$$

First note that by introducing some additional variables and linear constraints then it can be assumed that no two Q matrices has a nonzero in the same positions. Moreover, by introducing some additional linear constraints and by rescaling the variables then it can be assumed that all the elements in the matrices Q belong to the set $\{-1, 0, 1\}$.

We will therefore assume this is the case. After this transformation it is checked whether any of the quadratic inequalities falls into one of the following cases.

Case 1: Q has the form (59). Moreover, assume all the diagonal elements in Q are positive, $a = 0$, and $b \leq 0$. In this case the constraint (58) can be written as

$$\begin{array}{rcll} u & = & \sqrt{-2b}, & \\ x^T Qx & \leq & u^2, & u \geq 0, \end{array}$$

where u is an additional variable.

Case 2: Q has the form (59) and all the diagonal elements are positive. In this case the constraint (58) can be written as

$$\begin{array}{rcl} u + a^T x + b & = & 0, \\ x^T Q x & \leq & 2uv, \quad u, v \geq 0, \\ v & = & 1, \end{array}$$

where u and v are two additional variables.

Case 3: Q has the form (60) and all the diagonal elements are positive except the jth element i.e., $q_{jj} < 0$. Moreover, the assumptions $x_j \geq 0$, $a = 0$, and $b \geq 0$ should be satisfied. In this case the constraint (58) can be written as

$$\begin{array}{rcl} u & = & \sqrt{2b} \\ (x^T Q^i x - q_{jj} x_j^2) + u^2 & \leq & -q_{jj} x_j^2, \end{array}$$

where u is an additional variable. If $b = 0$, then it is not necessary to introduce u.

Case 4: Q has the form (60) and all the diagonal elements are positive and $q_{ij} < 0$. Moreover,the assumptions $x_i, x_j \geq 0$, $a = 0$, and $b \geq 0$ should be satisfied. In this case the constraint (58) can be written as

$$\begin{array}{rcl} u & = & \sqrt{2b}, \\ x^T Q x - 2q_{ij} x_j x_i + u^2 & \leq & -2q_{ij} x_i x_j, \end{array}$$

where u is an additional variable.

Observe that it is computationally cheap to check whether one of the four cases occurs for a particular quadratic inequality.

In all four cases a modified quadratic inequality is obtained possibly after the inclusion of some linear variables and constraints. Moreover, these quadratic inequalities can be represented by quadratic cones implying a conic quadratic optimization problem in the required form is obtained. (This may involve the introduction of some additional linear constraints to make the appropriate variables in the cone free i.e. instead of letting x_i be a member of a cone, then a constraint having the form $x_i = x_j$ is introduced and we let x_j be a member of the cone.)

Hence, in our implementation we assume that the user specifies the conic quadratic optimization problem using linear constraints and quadratic inequalities. Moreover, it is implemented such that the system checks whether the quadratic inequalities can be converted to conic quadratic form and this conversion is automatic if the quadratic inequalities satisfy the weak requirements discussed above.

7.2 Presolving the problem

Before the problem is optimized it is preprocessed to remove obvious redundancies using most of the techniques presented in [4]. For example fixed variables are removed, obviously redundant constraints are removed, linear dependencies in A are removed. Finally, some of the linear free variables are substituted out of the problem.

8 Computational results

A complete algorithm for solution of conic quadratic optimization problems has now been specified and we will now turn our attention to evaluating the practical efficiency of the presented algorithm.

The algorithm has been implemented in the programming language `C` and is linked to the MOSEK optimizer[3]. During the computational testing all the algorithmic parameters are held constant at the values shown in Table 1. In the computational results reported, then the method of multiple correctors is not employed because it gave only rise to insignificant saving in the computation time.

The computational test is performed on a 333MHZ PII based PC having 256MB of RAM. The operating system used is Windows NT 4.

In Table 2 the test problems are shown along with the size of the problems before and after the presolve procedure has been applied to the problems. The test problem comes from different sources. The problems belonging to the `nb*`, `nql*`, `qssp*`, and `*socp` families are all DIMACS Challenge problems [24]. The `dttd*` family of problems are multi load truss topology design problems, see [11, p. 117]. The `traffic*` problems arises from a model developed by C. Roos to study a traffic phenomena. The remaining problems have been obtained by the authors from various sources. The problems in

[3]See http://www.mosek.com

Constant	Value	Section
β	10^{-8}	3.1
δ	0.5	4.2
$\bar{\rho}_P$	10^{-8}	6.2
$\bar{\rho}_D$	10^{-8}	6.2
$\bar{\rho}_A$	10^{-8}	6.2
$\bar{\rho}_G$	10^{-8}	6.2
$\bar{\rho}_I$	10^{-10}	6.2
$\bar{\rho}_\mu$	10^{-10}	6.2

Table 1: Algorithmic parameters.

the `nql*` and `qssp*` families have previously been solved in [10] and are dual problems of minimum sum of norms problems.

It is evident from Table 2 that the presolve procedure in some cases is effective at reducing the problem size. In particular the `nql*` and `traffic*` models are reduced significantly by the presolve procedure.

The purpose of the subsequent Table 3 is to show various optimizer related statistics i.e. the size of the problems actually solved and performance related statistics. Recall that in some cases additional constraints and variables are added to the problems to state them in the required conic quadratic form. The first two columns of Table 3 show the number of constraints and the number of cones in each problem. Next the total number of variables, the number variables which has both a finite lower and upper bound, and the number of variables which is member of a cone are shown. Finally, the number interior-point iterations performed to optimize the problems, the time spend in the interior-point optimizer, and the total solution time are shown.

The main conclusion that can be drawn from Table 3 is even though some of the problems are large, then the total number of interior-point iterations required to solve each problem are small. An important observation is that the number of iterations tend to grow slowly with the problem size. This can for instance be observed for the `nql*` and `qssp*` problems.

Finally, in Table 4 we show feasibility and optimality related measures. The columns primal and dual feasibility report the numerator in $\rho_P^{(*)}$ and $\rho_D^{(*)}$ respectively. The first "primal objective" and "Sig fig." columns show the value $c^T x^*$ the number figures that are identical in the optimal primal and dual objective values as reported by the interior-point optimizer. In all cases those numbers demonstrate that the required accuracy is achieved and about 8 figures in the reported primal and dual objective values are identical. The final two columns of Table 4 shows the optimal primal objective value reported to the user specified model and the corresponding number of significant figures. Recall, that the optimization problems are specified using quadratic inequalities but is solved on conic form. Hence, the solution to the conic model has to be converted to a primal and dual solution to the model based on the quadratic inequalities.

In general it can be seen that the primal objective value reported by the interior-point optimizer and the one corresponding to the converted primal solution is almost identical. However, the number of significant figures is not so high. This indicates the dual solution looses some of the accuracy when it is converted.

9 Conclusion

The present work discusses a primal-dual interior-point method designed to solve large-scale sparse conic quadratic optimization problems. The main theoretical features of the algorithm is that it employs the Nesterov-Todd search direction and the homogeneous model. Moreover, the algorithm has been extended with a Mehrotra predicter-corrector scheme, treats the rotated quadratic cone without introducing additional variables and constraints and employs structure and sparsity exploiting linear algebra.

The presented computational results indicates that the suggested algorithm is capable of computing accurate solutions to very large sparse conic quadratic optimization problems in a fairly low amount of time.

Although the proposed algorithm works well then, some work is left for the future. Particularly it might be possible to invent a heuristic for computing a good starting point which might improve the efficiency of the algorithm. Also it should be possible to specify the problem directly in conic form instead of using the implicit form based on quadratic inequalities. This would make it easy to report an accurate optimal dual solution.

Acknowledgement: We would like to thank K. D. Andersen and A. Nemirovskii for helping us during the collection of the test problems.

A Appendix

Definition A.1 *Let K be a pointed and closed convex cone, then K is self-dual if*

$$K = K_*$$

and homogeneous if for any $x, s \in int(K)$ we have

$$\exists B \in R^{n\times n}: \; B(K) = K, \; Bx = s.$$

Self-dual and homogeneous cones have been studied extensively in the literature and is called self-scaled by Nesterov and Todd.

Proof of Lemma 3.1:

We first prove $X^i S^i = 0$ implies $(x^i)^T s^i = 0$. Observe that

$$\begin{array}{rcl} e^T X S e & = & \sum_{i=1}^{n} (e^i)^T X^i S^i e^i \\ & = & \sum_{i=1}^{k} (T^i x^i)^T T^i s^i \\ & = & x^T s. \end{array} \tag{61}$$

This implies that any solution satisfying (20) is also a complementary solution. Next we prove if $(x^i)^T s^i = 0$, then $X^i S^i e^i = 0$. In explicit form the complementarity conditions can be stated as

$$\begin{array}{rcl} (x^i)^T s^i & = & 0, \\ (T^i x^i)_1 (T^i s^i)_{2:n} + (T^i s^i)_1 (T^i x^i)_{2:n} & = & 0. \end{array}$$

Note that $T^i x^i, T^i s^i \in K^q$. This implies if either $(T^i x^i)_1 = 0$ or $(T^i s^i)_1 = 0$ then (20) is true as claimed. Therefore, assume this is not the case. Since x and s are complementary then

$$\begin{array}{rcl} 0 & = & x^T s \\ & = & \sum_{i=1}^{k} (T^i x^i)^T T^i s^i \\ & \geq & \sum_{i=1}^{k} (T^i x^i)_1 (T^i s^i)_1 - \|(T^i x^i)_{2:n}\| \, \|(T^i s^i)_{2:n}\| \\ & \geq & \sum_{i=1}^{k} \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i} \\ & \geq & 0 \end{array}$$

The first inequality follows from the Cauchy-Schwartz inequality and the second inequality follows from $T^i x^i, T^i s^i \in K^q$. This implies that both $(x^i)^T Q^i x^i = 0$ and $(s^i)^T Q^i s^i = 0$ are the case. Moreover, we conclude that

$$(T^i x^i)_1 (T^i s^i)_1 = \left\|(T^i x^i)_{2:n}\right\| \left\|(T^i s^i)_{2:n}\right\|.$$

However, this can only be the case if

$$\exists \alpha : \; (T^i x^i)_{2:n} = \alpha (T^i s^i)_{2:n}$$

for some $\alpha \in R$. Therefore, given the assumptions we have

$$\begin{array}{rcl} 0 & = & (x^i)^T s^i \\ & = & (T^i x^i)_1 (T^i s^i)_1 + \alpha \left\|(T^i s^i)_{2:n^i}\right\|^2 \end{array}$$

and

$$\alpha = -\frac{(T^i x^i)_1}{(T^i s^i)_1}$$

implying the complementarity conditions (20) are satisfied.

Proof of Lemma 3.3:

i), ii), and iii) follows immediately from the definition of a scaling. iv) is proved next. In the case K^i is R_+ then the statement is obviously true. In the case K^i is the quadratic cone then due to $W^i Q^i W^i = I$ we have that

$$w_1^2 - \|w_{2:n}\|^2 = 1,$$

where w denotes the first row of W^i because. This implies

$$\begin{array}{rcl} \bar{x}_i^1 & = & (e^i)^T \bar{x}^i \\ & = & (e^i)^T \Theta W^i x^i \\ & = & \theta_i (w_1 x_1^i + w_{2:n}^T x_{2:n}^i) \\ & \geq & \theta_i (w_1 x_1^i - \|w_{2:n^i}\| \, \|x_{2:n}^i\|) \\ & = & \theta_i (\sqrt{1 + \|w_{2:n^i}\|^2} x_1^i) \\ & \geq & \theta_i x_1^i \\ & \geq & 0 \end{array}$$

and

$$(\bar{x}^i)^T Q^i \bar{x}^i = \theta_i (x^i)^T Q^i x^i \geq 0.$$

Hence, $x^i \in K^i$ implies $\bar{x}^i \in K^i$ for the quadratic cone. Similarly, it easy to verify $x^i \in \text{int}(K^i)$ implies $\bar{x}^i \in \text{int}(K^i)$. Now assume K^i is a rotated quadratic cone and $x^i, s^i \in K^i$. Let $\hat{x}^i := T^i x^i$ and $\hat{s}^i := T^i s^i$ then $\hat{x}^i, \hat{s}^i \in K^q$. Therefore, a scaling $\hat{\theta}_i \hat{W}^i$ exist such that

$$\hat{s}^i = T^i s^i = (\hat{\theta}_i \hat{W}^i)^2 T^i x^i = (\hat{\theta}_i \hat{W}^i)^2 \hat{x}^i,$$

where

$$\begin{aligned} \bar{\theta}_i^2 &= \sqrt{\tfrac{(T^i s^i)^T \hat{Q}^i T^i s^i}{(T^i x^i)^T \hat{Q}^i T^i x^i}} \\ &= \theta_i^2. \end{aligned}$$

This implies

$$s^i = \hat{\theta}_i^2 T^i \hat{(W^i)}^2 T^i x^i$$

which shows $W^i = T^i \hat{W}^i T^i$. We know that $\hat{\theta}_i \hat{W}^i T^i x^i \in K^q$ and hence $\hat{\theta}_i T^i \hat{W}^i T^i x^i = \theta_i W^i x^i \in K^r$. vi) Follows from v) and the fact $\sqrt{x^T Q x s^T Q s} = \sqrt{\bar{x}^T Q \bar{x} \bar{s}^T Q \bar{s}}$.

Proof of Lemma 4.2:

(30) follows immediately from Lemma 3.3. It is trivial to compute W^i in the case K^i is R_+. Next assume K^i is a quadratic cone. First define

$$\tilde{w}_1 := w_1^i \quad \text{and} \quad \tilde{w}_2 := w_{2:n^i}^i$$

then

$$\begin{aligned} W^i W^i &= \begin{bmatrix} \tilde{w}_1 & \tilde{w}_2^T \\ \tilde{w}_2 & I + \frac{\tilde{w}_2 \tilde{w}_2^T}{1+\tilde{w}_1} \end{bmatrix} \begin{bmatrix} \tilde{w}_1 & \tilde{w}_2^T \\ \tilde{w}_2 & I + \frac{\tilde{w}_2 \tilde{w}_2^T}{1+\tilde{w}_1} \end{bmatrix}^T \\ &= \begin{bmatrix} \|\tilde{w}\|^2 & \left(1 + \tilde{w}_1 + \frac{\|\tilde{w}_2\|^2}{1+\tilde{w}_1}\right) \tilde{w}_2^T \\ \left(1 + \tilde{w}_1 + \frac{\|\tilde{w}_2\|^2}{1+\tilde{w}_1}\right) \tilde{w}_2 & \tilde{w}_2 \tilde{w}_2^T + \left(I + \frac{\tilde{w}_2 \tilde{w}_2^T}{1+\tilde{w}_1}\right)^2 \end{bmatrix} \\ &= -Q^i + 2w^i (w^i)^T, \end{aligned} \tag{62}$$

because

$$(w_1^i)^2 - \left\| w_{2:n^i}^i \right\|^2 = 1$$

follows from the definition of Q^i and the fact $W^i Q^i W^i = Q^i$. When (62) is combined with (29) one has

$$s^i = \theta_i^2 (-Q^i + 2w^i (w^i)^T) x^i$$

and

$$(x^i)^T s^i = \theta_i^2(-(x^i)^T Q^i x^i + 2((w^i)^T x^i)^2).$$

Therefore,

$$\begin{array}{rcl} 2\theta_i^2((w^i)^T x^i)^2 & = & (x^i)^T s^i + \theta_i^2 (x^i)^T Q^i x^i \\ & = & (x^i)^T s^i + \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i} \end{array}$$

and then

$$w^i = \frac{\theta_i^{-1} s^i + \theta_i Q^i x^i}{\sqrt{2}\sqrt{(x^i)^T s^i + \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i}}}.$$

Clearly,

$$(x^i)^T s^i + \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i} > 0$$

when $x^i, s^i \in \operatorname{int}(K)^i$. Now assume K^i is the rotated quadratic cone. Let $\hat{x}^i := T^i x^i$ and $\hat{s}^i := T^i s^i$ then $\hat{x}^i, \hat{s}^i \in \operatorname{int}(K^q)$. Moreover, define

$$\hat{Q}^i := T^i Q^i T^i$$

and

$$\hat{W}^i := T^i W^i T^i.$$

Since $T^i T^i = I$ and $W^i Q^i W^i = Q^i$ then $\bar{W}^i \hat{Q}^i \hat{W}^i = \hat{Q}^i$. Further by definition we have that $s^i = \theta_i^2 (W^i)^2 x^i$ which implies

$$\begin{array}{rcl} \hat{s}^i & = & T^i s^i \\ & = & \theta_i^2 T^i (W^i)^2 x^i \\ & = & \hat{\theta}_i^2 (T^i W^i T^i)^2 T^i x^i \\ & = & \hat{\theta}_i^2 (\hat{W}^i)^2 \hat{x}^i \end{array} \tag{63}$$

because

$$\begin{array}{rcl} \hat{\theta}_i^2 & = & \sqrt{\frac{(T^i s^i)^T \bar{Q}^i T^i s^i}{(T^i x^i)^T \bar{Q}^i T^i x^i}} \\ & = & \theta_i^2. \end{array}$$

Now $\hat{x}^i, \hat{s}^i \in \operatorname{int}(K^q)$ which implies we can use relation (31) to compute the scaling $\hat{W}^i$ in (63). Therefore,

$$\begin{array}{rcl} \hat{w}^i & = & \frac{\hat{\theta}_i^{-1} \hat{s}^i + \hat{\theta}_i \hat{Q}^i \hat{x}^i}{\sqrt{2}\sqrt{(\hat{x}^i)^T \hat{s}^i + \sqrt{(\hat{x}^i)^T \hat{Q}^i \hat{x}^i (T^i s^i)^T \hat{Q}^i T^i s^i}}} \\ & = & \frac{\theta_i^{-1} T^i s^i + \theta_i T^i Q^i x^i}{\sqrt{2}\sqrt{(x^i)^T s^i + \sqrt{(x^i)^T Q^i x^i (s^i)^T Q^i s^i}}} \\ & = & T^i w^i \end{array}$$

and

$$\begin{aligned}
\hat{W}^i &= -\hat{Q}^i + \tfrac{(e_1^i+\hat{w}^i)(e_1^i+\hat{w}^i)^T}{1+\hat{w}^i} \\
&= T^i \left(-Q^i + \tfrac{(T^i e_1^i + w^i)(T^i e_1^i + w^i)^T}{1+(e_1^i)^T T^i w^i} \right) T^i \\
&= T^i W^i T^i
\end{aligned}$$

from which (34) follows.

References

[1] I. Adler and F. Alizadeh. Primal-dual interior point algorithms for convex quadratically constrained and semidefinite optimization problems. Technical Report RRR-111-95, RUTCOR, Rutgers Center for Operations Research, P.O. Box 5062, New Brunswick, New Jersey, 1995.

[2] F. Alizadeh and S. H. Schmieta. Optimization with semidefinite, quadratic and linear constraints. Technical Report RRR 23-97, RUTCOR, Rutgers Center for Operations Research, P.O. Box 5062, New Brunswick, New Jersey, November 1997.

[3] S. H. Schmieta F. Alizadeh. Associative algebras, symmetric cones and polynomial time interior point algorithms. Technical Report RRR 17-98, RUTCOR, Rutgers Center for Operations Research, P.O. Box 5062, New Brunswick, New Jersey, June 1998.

[4] E. D. Andersen and K. D. Andersen. Presolving in linear programming. *Math. Programming*, 71(2):221–245, 1995.

[5] E. D. Andersen and K. D. Andersen. A parallel interior-point based linear programming solver for shared-memory multiprocessor computers: A case study based on the XPRESS LP solver. Technical Report CORE DP 9808, CORE, UCL, Belgium, 1997.

[6] E. D. Andersen and K. D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High performance optimization*, pages 197–232, 2000.

[7] E. D. Andersen, J. Gondzio, Cs. Mészáros, and X. Xu. Implementation of interior point methods for large scale linear programming. In T. Terlaky, editor, *Interior-point methods of mathematical programming*, pages 189–252. Kluwer Academic Publishers, 1996.

[8] K. D. Andersen. *Minimizing a Sum of Norms (Large Scale solutions of symmetric positive definite linear systems)*. PhD thesis, Odense University, 1995.

[9] K. D. Andersen. QCOPT a large scale interior-point code for solving quadratically constrained quadratic problems on self-scaled cone form. 1997.

[10] K. D. Andersen, E. Christiansen, and M. L. Overton. Computing limit loads by minimizing a sum of norms. *SIAM Journal on Scientific Computing*, 19(2), March 1998.

[11] A. Ben-Tal and A Nemirovski. Convex optimization in engineering: Modeling, analysis, algorithms. 1999.

[12] E. de Klerk, C. Roos, and T. Terlaky. Initialization in semidefinite programming via a self–dual, skew–symmetric embedding. *OR Letters*, 20:213–221, 1997.

[13] A. J. Goldman and A. W. Tucker. Theory of linear programming. In H. W. Kuhn and A. W. Tucker, editors, *Linear Inequalities and related Systems*, pages 53–97, Princeton, New Jersey, 1956. Princeton University Press.

[14] J. Gondzio. Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications*, 6:137–156, 1996.

[15] O. Güler. Barrier functions in interior-point methods. *Math. Oper. Res.*, 21:860–885, 1996.

[16] M. Kojima, S. Mizuno, and A. Yoshise. A primal-dual interior point algorithm for linear programming. In N. Megiddo, editor, *Progress in Mathematical Programming: Interior-Point Algorithms and Related Methods*, pages 29–47. Springer Verlag, Berlin, 1989.

[17] M. S. Lobo, L. Vanderberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra Appl.*, pages 193–228, November 1998.

[18] I. J. Lustig, R. E. Marsten, and D. F. Shanno. Interior point methods for linear programming: Computational state of the art. *ORSA J. on Comput.*, 6(1):1–15, 1994.

[19] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM J. on Optim.*, 2(4):575–601, 1992.

[20] R. D. C. Monteiro and I. Adler. Interior path following primal-dual algorithms. Part I: Linear programming. *Math. Programming*, 44:27–41, 1989.

[21] R. D. C. Monteiro and T. Tsuchiya. Polynomial convergence of primal-dual algorithms for the second-order cone program based on the MZ-family of directions. *Math. Programming*, 88(1):61–83, 2000.

[22] Y. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Math. Oper. Res.*, 22(1):1–42, February 1997.

[23] Yu. Nesterov, M. J. Todd, and Y. Ye. Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems. *Math. Programming*, 84(2):227–267, February 1999.

[24] G. Pataki and S. Schmieta. The DIMACS library of semidefinte-quadratic-linearprograms. Technical report, Computational Optimization Research Center, Columbia University, November 1999.

[25] J. F. Sturm. *Primal-dual interior point approach to semidefinite programming.* PhD thesis, Tinbergen Institute, Erasmus University Rotterdam, 1997.

[26] J. F. Sturm. SeDuMi 1.02, a MATLAB toolbox for optimizing over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[27] T. Tsuchiya. A polynomial primal-dual path-following algorithm for second-order cone programming. Technical report, The Institute of Statistical Mathematics, Tokyo, Japan, October 1997.

[28] X. Xu, P. -F. Hung, and Y. Ye. A simplified homogeneous and self-dual linear programming algorithm and its implementation. *Annals of Operations Research*, 62:151–171, 1996.

Name	Before presolve Constraints	Variables	Nz(A)	After presolve Constraints	Variables	Nz(a)
drttd-11-11-2	14853	36026	70017	14853	36026	70017
drttd-11-7-2	6093	14526	27757	6093	14526	27757
drttd-11-7-7	21323	43576	89887	21323	43576	89887
drttd-21-7-1	10992	32131	58661	10991	32130	47950
drttd-21-7-2	21983	53551	106612	21983	53551	106612
drttd-9-9-2	6699	16021	30350	6699	16021	30350
nb	916	2383	191519	916	2381	191293
nb_L1	1708	3176	192312	915	2381	190500
nb_L2	962	4195	402285	961	4192	402050
nb_L2_bessel	962	2641	208817	961	2638	208582
nql21	1820	1765	8287	1720	1666	8287
nql30	3680	3601	16969	3567	3489	16969
nql60	14560	14401	68139	14391	14233	68139
nql90	32640	32401	153509	32336	32098	153509
nql180	130080	129601	614819	129491	129013	614819
qssp30	3691	5674	34959	3690	5673	34950
qssp60	14581	22144	141909	14580	22143	141900
qssp90	32671	49414	320859	32670	49413	320850
qssp180	130141	196024	1289709	130140	196023	1289700
than1	285560	264557	944944	285560	264557	944944
than2	5364	4861	16884	5364	4861	16884
than3	301	520	1524	301	520	1524
than4	520	541	1884	520	541	1884
than5	412064	374461	1320564	412064	374461	1320564
traffic-12	2277	1342	4383	887	876	4380
traffic-24	4593	2722	8955	1888	1865	8955
traffic-36	6909	4102	13527	2899	2864	13527
sched_100_100_orig	8340	18240	104902	8340	18240	104902
sched_100_100_scaled	8338	18238	114899	8338	18238	114899
sched_100_50_orig	4846	9746	55291	4846	9746	55291
sched_100_50_scaled	4844	9744	60288	4844	9744	60288
sched_200_100_orig	18089	37889	260503	18089	37889	260503
sched_200_100_scaled	18087	37887	280500	18087	37887	280500
sched_50_50_orig	2529	4979	25488	2529	4979	25488
sched_50_50_scaled	2527	4977	27985	2527	4977	27985
Sum	1235543	1458057	7269897	1225363	1453418	7256639

Table 2: The test problems.

	Constraints		Variables			Optimizer		
							Time (s)	
Name	Linear	Cone	Total	Upper	Cone	Itera-tions	Interior-point	Total
drttd-11-11-2	7648	14410	43231	0	43230	42	79.0	79.0
drttd-11-7-2	3188	5810	17431	0	17430	33	22.2	22.2
drttd-11-7-7	18418	20335	61006	0	61005	35	224.1	224.2
drttd-21-7-1	281	10710	32130	0	32130	32	30.4	30.4
drttd-21-7-2	11273	21420	64261	0	64260	44	127.9	127.9
drttd-9-9-2	3495	6408	19225	0	19224	42	30.7	30.7
nb	123	793	2381	0	2379	19	21.0	21.0
nb_L1	915	793	3174	0	2379	19	24.8	24.8
nb_L2	122	839	4193	0	4191	22	65.9	65.9
nb_L2_bessel	122	839	2639	0	2637	14	16.0	16.0
nql21	1279	441	2107	0	1323	16	1.8	1.8
nql30	2667	900	4389	0	2700	18	5.2	5.2
nql60	10791	3600	17833	0	10800	19	29.4	29.4
nql90	24236	8100	40198	0	24300	21	82.3	82.3
nql180	97091	32400	161413	0	97200	23	521.6	521.7
qssp30	1799	1891	7564	0	7564	16	6.7	6.7
qssp60	7199	7381	29524	0	29524	20	45.3	45.3
qssp90	16199	16471	65884	0	65884	19	117.8	117.9
qssp180	64799	65341	261364	0	261364	19	771.2	771.3
than1	229864	55696	320253	0	278480	18	561.3	561.4
than2	4068	1296	6157	0	5184	12	3.5	3.5
than3	181	120	520	0	480	13	0.2	0.2
than4	400	120	661	0	480	13	0.3	0.3
than5	312208	99856	474317	0	399424	17	824.3	824.5
traffic-12	1305	418	2141	825	1254	30	2.7	2.7
traffic-24	2762	874	4510	1725	2622	37	8.1	8.1
traffic-36	4229	1330	6889	2625	3990	35	12.2	12.2
sched_100_100_orig	8338	2	18240	0	8238	30	25.2	25.2
sched_100_100_scaled	8337	1	18238	0	8236	37	31.7	31.8
sched_100_50_orig	4844	2	9746	0	4744	26	11.9	11.9
sched_100_50_scaled	4843	1	9744	0	4742	25	12.0	12.1
sched_200_100_orig	18087	2	37889	0	17887	31	69.6	69.6
sched_200_100_scaled	18086	1	37887	0	17885	32	74.3	74.3
sched_50_50_orig	2527	2	4979	0	2477	28	5.4	5.4
sched_50_50_scaled	2526	1	4977	0	2475	19	4.2	4.2
Sum	894250	378604	1797095	5175	1508122	876	3870.3	3871.0

Table 3: Problem and optimizer results.

	Optimizer				Final	
	Feasibility		Primal	Sig.	Primal	Sig.
Name	primal	dual	objective	fig.	objective	fig.
drttd-11-11-2	1.2e-008	1.7e-010	7.61821404e+003	9	7.61821404e+003	5
drttd-11-7-2	1.2e-007	2.5e-009	1.14361939e+003	9	1.14361939e+003	5
drttd-11-7-7	4.3e-008	1.1e-009	8.57614313e+003	9	8.57614313e+003	5
drttd-21-7-1	1.6e-008	1.6e-010	1.40452440e+004	9	1.40452440e+004	5
drttd-21-7-2	3.8e-008	4.3e-010	7.59987339e+003	9	7.59987339e+003	5
drttd-9-9-2	7.3e-009	1.5e-010	5.78124166e+003	9	5.78124166e+003	5
nb	4.8e-006	1.7e-007	-5.07030944e-002	10	-5.07030944e-002	9
nb_L1	2.7e-007	2.4e-008	-1.30122697e+001	9	-1.30122697e+001	8
nb_L2	2.8e-006	1.7e-007	-1.62897176e+000	8	-1.62897176e+000	8
nb_L2_bessel	6.4e-008	1.0e-008	-1.02569500e-001	8	-1.02569500e-001	8
nql21	1.9e-013	1.5e-007	-9.55221085e-001	9	-9.55221085e-001	6
nql30	2.9e-011	1.9e-007	-9.46026849e-001	9	-9.46026849e-001	6
nql60	2.3e-010	3.8e-007	-9.35048150e-001	9	-9.35048150e-001	5
nql90	8.4e-011	4.8e-007	-9.31375780e-001	9	-9.31375780e-001	5
nql180	2.8e-009	1.4e-006	-9.27698253e-001	9	-9.27698253e-001	4
qssp30	3.5e-014	2.2e-008	-6.49667527e+000	9	-6.49667527e+000	8
qssp60	1.6e-013	2.4e-007	-6.56269681e+000	10	-6.56269681e+000	6
qssp90	3.3e-013	9.9e-008	-6.59439558e+000	11	-6.59439558e+000	7
qssp180	6.5e-011	8.4e-007	-6.63951191e+000	14	-6.63951191e+000	5
than1	1.6e-010	2.2e-006	-5.59410181e+000	13	-5.59410181e+000	6
than2	5.1e-010	2.5e-007	-5.30444254e-001	11	-5.30444254e-001	7
than3	4.8e-010	1.4e-011	7.76311624e-001	10	7.76311624e-001	10
than4	1.7e-010	5.0e-009	-7.76311611e-001	10	-7.76311611e-001	8
than5	7.6e-009	2.8e-006	-4.63823105e-001	14	-4.63823105e-001	5
traffic-12	1.0e-006	1.2e-007	8.63441371e+002	9	8.63441371e+002	5
traffic-24	6.0e-007	5.4e-008	2.68377252e+003	9	2.68377252e+003	4
traffic-36	4.6e-007	3.5e-008	5.39022850e+003	9	5.39022850e+003	4
sched_100_100_orig	1.1e+000	1.4e-008	7.17367643e+005	9	7.17367643e+005	9
sched_100_100_scaled	4.7e-002	2.7e-008	2.73314579e+001	9	2.73314579e+001	7
sched_100_50_orig	1.0e-001	3.2e-007	1.81889861e+005	9	1.81889861e+005	8
sched_100_50_scaled	3.0e-002	1.4e-007	6.71662899e+001	9	6.71662899e+001	7
sched_200_100_orig	2.4e-001	3.1e-007	1.41359802e+005	9	1.41359802e+005	7
sched_200_100_scaled	9.7e-002	2.3e-007	5.18124709e+001	10	5.18124709e+001	8
sched_50_50_orig	3.5e-004	1.8e-009	2.66729911e+004	9	2.66729911e+004	8
sched_50_50_scaled	2.0e-002	1.6e-007	7.85203845e+000	10	7.85203845e+000	10

Table 4: Feasibility measures and objective values.