On Implementing a Homogeneous Interior-Point Algorithm for Nonsymmetric Conic Optimization

Skajaa, Anders; Jørgensen, John Bagterp; Hansen, Per Christian

Publication date: 2011

Document Version: Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
ON IMPLEMENTING A HOMOGENEOUS INTERIOR-POINT ALGORITHM FOR NONSYMMETRIC CONIC OPTIMIZATION

Anders Skajaa∗, John Bagterp Jørgensen∗ and Per Christian Hansen∗

January, 2011

Abstract

Based on earlier work by Nesterov, an implementation of a homogeneous infeasible-start interior-point algorithm for solving nonsymmetric conic optimization problems is presented. Starting each iteration from (the vicinity of) the central path, the method computes (nearly) primal-dual symmetric approximate tangent directions followed by a purely primal centering procedure to locate the next central primal-dual point. Features of the algorithm include that it makes use only of the primal barrier function, that it is able to detect infeasibilities in the problem and that no phase-I method is needed. The method further employs quasi-Newton updating both to generate (pseudo) higher order directions and to reduce the number of factorizations needed in the centering process while still retaining the ability to exploit sparsity. Extensive and promising computational results are presented for the $p$-cone problem, the facility location problem, entropy problems and geometric programs; all formulated as nonsymmetric conic optimization problems.

Keywords: convex optimization, nonsymmetric, conic optimization, homogeneous model, infeasible-start, interior-point algorithm.

1 Introduction

In this paper we are concerned with conic optimization problem pairs of the form

\[
\begin{align*}
\text{PRIMAL} & \quad \left\{ \begin{array}{ll}
\min_x & c^T x \\
\text{s.t.} & Ax = b \\
x \in K
\end{array} \right. \\
\text{DUAL} & \quad \left\{ \begin{array}{ll}
\max_{y,s} & b^T y \\
\text{s.t.} & A^T y + s = c \\
 & s \in K^*, \quad y \in \mathbb{R}^m
\end{array} \right.
\end{align*}
\]

(1)

where $x, c \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, K \subseteq \mathbb{R}^n$ is a proper cone (i.e. it is convex, pointed, closed and has nonempty interior) and $K^* = \{ s \in \mathbb{R}^n : s^T x \geq 0, \forall x \in K \}$, its dual cone, which is also proper. We are further assuming that $m \leq n$ and that $A$ has full row-rank.

These problems are well described and many interior point methods for different variants of these problems exist, see for example [4, 5, 25, 21, 14].

When replacing $K$ in (1) with $\mathbb{R}_+^n$, we obtain the simplest case: Linear Programming (LP). Solution methods for LP have been studied for long in different settings and until the emergence of interior point methods (IPMs), the most prominent method was the simplex method, developed by Dantzig in the 1940s. The introduction of IPMs is usually ascribed to Karmarkar [12] in 1984 and since then, research in the area has been extensive.

In [17] it was studied how to extend the ideas of IPMs to the nonlinear case. The papers [18, 19] showed that problems of the type (1) are efficiently solvable using symmetric primal-dual IPMs when $K$ admits a self-scaled barrier function $F: K^* \to \mathbb{R}$. In [11], Güler demonstrated that such cones are limited to those that are homogeneous and self-dual; a class that encompasses just five cones of which only

∗Department of Informatics and Mathematical Modelling, Technical University of Denmark
three are interesting for optimization. Specifically: the positive orthant (leading to l_p), the Lorentz cone (leading to second order cone programming) and the positive semidefinite cone (leading to semidefinite programming).

These three self-scaled cones allow for modelling of a great variety of constraints and with the existence of a unique scaling point [18, 19] for each primal-dual pair, symmetric primal-dual interior-point methods (PDIPM) can be constructed and very efficiently implemented [1, 2, 23].

However many constraints do not fall in this class. Examples include entropy type constraints: \( x \log x \leq t \), \( p \)-cone constraints: \( \|x\|_p \leq t \), and constraints arising in geometric programming [5]. Some of these constraints can be modelled using self-scaled cones, but this usually requires the introduction of many extra variables and constraints [4].

Theoretically, one can solve problems involving any convex constraint using a purely primal barrier method and still obtain an algorithm with the best known complexity. However, such an algorithm is known to be practically less efficient than a PDIPM. Other approaches are also possible and special algorithms for certain problems exist [25, 27]. An approach known to be effective for general convex problems is to solve the monotone complementarity problem, see for example [3].

However, it may be beneficial to model non-symmetric constraints more directly using non-self-scaled cones, such as the power cone or the exponential cone. This approach was employed by Nesterov in [16].

He proposed a method that mimics the ideas of a PDIPM for symmetric cones by splitting each iteration into two phases. First, a pure primal centering (or correction) phase is used to find a primal central point \( x \) and a scaling point \( w \). These points are used to compute a feasible dual point \( s \) such that an exact scaling relation is satisfied: \( s = \nabla^2 F(w)x \). Second, a true symmetric primal-dual affine-scaling step (or prediction step) is taken.

Extending the algorithm of [16], we propose in this paper a primal-dual interior point algorithm for a homogeneous model of (1). This approach has been succesful for self-scaled cones [26, 2, 22] because it implies the following desirable properties: Ability to detect infeasibility and ease of finding a suitable starting point, removing the need for a phase I method. In addition to this, we suggest using BFGS-updating of the Hessian of the barrier function to obtain a pseudo-second order approximate tangent direction (ATD) and to speed up the centering process.

For all problems that we consider, \( K \) will have the form \( K = K_1 \times \cdots \times K_K \) where each \( K_j \) is either a three-dimensional proper cone or \( \mathbb{R}_+ \). We assume that a logarithmically homogeneous self-concordant barrier function (LHSCB) \( F \) for \( K \), its gradient \( \nabla F \) and its Hessian \( \nabla^2 F \) are available and computable in polynomial time for all \( x \in K \).

The paper is organized as follows: We first list a few well known properties about LHSCBs and then we introduce notation and present our algorithm. We give a proof of convergence and then suggest some heuristic improvements to speed up convergence. Finally we present computational results for several problems.

2 Properties of LHSCB

We assume that \( F : K^\circ \to \mathbb{R} \) is a LHSCB for \( K \) with parameter \( \nu \). This means that for all \( x \in K^\circ \) and \( t > 0 \):

\[
F(tx) = F(x) - \nu \log t.
\]

It follows that the dual barrier of \( F \), denoted \( F^* \) and defined for \( s \in K^* \) by \( F^*(s) = \sup_{x \in K} \{-s^T x - F(x)\} \), is a LHSCB for the dual cone \( K^* \). Using the Hessians of \( F \) and \( F^* \), local norms on \( K \) and \( K^* \) can be defined:

\[
\|g\|_x = \sqrt{g^T (\nabla^2 F(x)) g}, \quad \text{for } g, x \in K \tag{2}
\]

\[
\|h\|_s^* = \sqrt{h^T (\nabla^2 F^*(s)) h}, \quad \text{for } h, s \in K^*. \tag{3}
\]

Properties of \( F \) and \( F^* \) include (see e.g. [18, 19]):

\[
\nabla^2 F(x)x = -\nabla F(x) \tag{4}
\]

\[
x \in K \Rightarrow -\nabla F(x) \in K^* \tag{5}
\]

\[
x^T \nabla F(x) = -\nu \tag{6}
\]

\[
\nabla^2 F^*(-\nabla F(x)) = \nabla^2 F(x)^{-1} \tag{7}
\]

\[
\|x\|_x = \nu. \tag{8}
\]
It is well known that the Dikin ellipsoid \([4]\) is feasible. That is:
\[ x \in \mathcal{K} \implies W(x) = \{ u, \|u - x\|_x \leq 1 \} \subseteq \mathcal{K} \quad (9) \]
and we have similarly for the dual cone that
\[ s \in \mathcal{K}^* \implies W^*(s) = \{ h, \|h - s\|_s^* \leq 1 \} \subseteq \mathcal{K}^* \quad (10) \]
If \( x \in \mathcal{K} \), we have by (5) and (10) that
\[ W^*(-\nabla F(x)) = \{ h, \|h + \nabla F(x)\|_{\nabla F(x)} \leq 1 \} \subseteq \mathcal{K}^* \quad (11) \]

\section{3 Notation}
Generally \( z \) denotes an aggregated variable \((x; \tau; y; s; \kappa) \in \mathbb{R}^{n+1+m+n+1}\) and \( w \) denotes an aggregated variable of the type \((x; \tau; y; \kappa)\). Semicolons indicate vertical stacking of column vectors.

Let
\[
G = \begin{pmatrix}
0 & A & -b \\
-A^T & 0 & c \\
b^T & -c^T & 0
\end{pmatrix}
\]
and notice that \( G \) is skew-symmetric: \( G = -G^T \). We will use the following abbreviations:
\[
\begin{align*}
r(z) = (r_P(z); r_D(z); r_G(z)) &= G(y; x; \tau) - (0; s; \kappa) \\
&= \begin{pmatrix} Ax - b \tau \\ -A^T y - s + c \tau \\ -c^T x + b^T y - \kappa \end{pmatrix} \\
r_A(z) &= \frac{|c^T x - b^T y|}{(\tau + |b^T y|)} \\
r_C(w, \mu) &= -A^T y + \mu \nabla F(x) + c \tau \\
\psi(z, \mu) &= s - \mu \nabla^2 F(x) \quad \mu(z) = (x^T s + \tau \kappa) / (\nu + 1).
\end{align*}
\]
When needed, we will use a superscript, e.g. \( z^{(k)} \), to denote iteration counter and a subscript, e.g. \( z_j \), to denote the \( j \)th element of the vector \( z \).

\section{4 Homogeneous algorithm}
For a parameter \( \mu > 0 \), the primal barrier problem corresponding to (1) is
\[
\min_x c^T x + \mu F(x), \quad \text{s.t.} \quad Ax = b \quad (13)
\]
and the KKT conditions of this problem can be stated as follows: If \( x \in \mathcal{K} \) is optimal for (13), then \( \exists s \in \mathcal{K}^*, y \in \mathbb{R}^m \) s.t. that
\[
\begin{align*}
Ax - b &= 0 \quad (14) \\
-A^T y - s + c &= 0 \quad (15) \\
s + \mu \nabla F(x) &= 0 \quad (16)
\end{align*}
\]
The points that satisfy (14)-(16) are known as the primal-dual central path and it is easily seen that they satisfy $b^Ty - c^Tx = x^Ts = \mu \nu$. The idea of the barrier method is to solve (13) for a decreasing sequence of $\mu$'s, thus obtaining a sequence of points with duality gap decreasing to zero and hence eventually being approximately optimal for (1).

In practice it is more efficient to take steps that are combinations of the direction approximately tangent to the central path (ATD) and the direction pointing towards the central path (the centering direction). After each step, $\mu$ is updated and the procedure repeated. Thus the iterates are guided by the central path, but not necessarily on it and such a method falls in the category of path-following methods or PDIPMs. We obtain different methods depending on the strategy for updating $\mu$ and how close the iterates are kept to the central path.

When $K$ is self-scaled, PDIPMs can compute symmetric search directions. Here symmetric refers to the search directions (and thus the iterates) being the same regardless of whether the roles of the primal and dual problems in (1) are interchanged [24]. Thus no particular emphasis is put on either the primal or the dual problem, which is a desirable feature of an algorithm.

In the case when $K$ is not self-scaled, this symmetry is lost. Instead, the idea of [16] is to split each iteration into two phases: 1. Prediction phase: Starting from a central point, compute a symmetric ATD and step in this direction and 2. Correction phase: compute a new central point using a purely primal iterative procedure.

In this paper, we propose solving the homogeneous model of problems (1) using the same two phases. We thus introduce two extra non-negative scalar variables $\tau$ and $\kappa$ and seek to find $z = (x; \tau; y; s; \kappa)$ such that

\begin{align}
Ax - b\tau &= 0 \tag{17} \\
-A^Ty - s + ct &= 0 \tag{18} \\
-c^Tx + b^Ty - \kappa &= 0 \tag{19}
\end{align}

$x \in K$, $s \in K^*$, $y \in \mathbb{R}^m$, $\tau \geq 0$, $\kappa \geq 0$.

\begin{align}
z &\in (x; \tau; y; s; \kappa) \text{ satisfies (17)-(20). Then} \nonumber \\
(i) &\text{ } z \text{ is complementary. That is: } (\nu + 1) \mu(z) = x^Ts + \tau \kappa = 0. \nonumber \\
(ii) &\text{ If } \tau > 0 \text{ then } (x, y, s)/\tau \text{ is optimal for (1).} \nonumber \\
(iii) &\text{ If } \kappa > 0 \text{ then, one or both of } b^Ty > 0 \text{ and } c^Tx < 0 \text{ hold. If the first holds, then (1) is primal-infeasible. If the second holds, then (1) is dual-infeasible.} \nonumber
\end{align}

Proof. (i) Observe that we can write (17)-(19) as $G(y; x; \tau) - (0; s; \kappa) = 0$. Pre-multiplying this equation by $(y; x; \tau)^T$ gives $x^Ts + \tau \kappa = 0$. (ii) $\tau > 0$ implies $\kappa = 0$ and hence by (19): $b^T(y/\tau) - c^T(x/\tau) = 0$. Further dividing (17) and (18) by $\tau$ we obtain (14) and (15) respectively. Thus $(x, y, s)/\tau$ is optimal for (1). (iii) If $\kappa > 0$ then $\tau = 0$ so $Ax = 0$ and $A^Ty + s = 0$. Further $c^Tx - b^Ty = -\kappa < 0$ so not both $c^Tx$ and $-b^Ty$ can be non-negative. Assume $-b^Ty < 0$. If (1) is primal-feasible then there exists $\bar{x} \in K$ such that $A\bar{x} = b$. But then $0 > -b^Ty = -\bar{x}^TA^Ty = \bar{x}^Ts \geq 0$, a contradiction. We can argue similarly if $c^Tx < 0$. \hfill \Box

Lemma 1 shows that any solution to (17)-(20) with $\tau + \kappa > 0$ provides either an optimal solution to our original problems (1) or a certificate of infeasibility of (one of) the original problems. See [13] for further details.

Let $z^{(0)}$ satisfy (20) and let $\mu^{(0)} = \mu(z^{(0)})$. Parametrized by $\gamma \in [0, 1]$, we will define the central path of problem (17)-(20) by

\begin{align}
\tau(z) &= \gamma \tau(z^{(0)}) \tag{21} \\
s &= -\gamma \mu^{(0)} \nabla F(x) \tag{22} \\
\tau \kappa &= \gamma \mu^{(0)}. \tag{23}
\end{align}

The central path connects the initial point ($\gamma = 1$) with a solution of (17)-(20) ($\gamma = 0$). The ATD $z'$ is then determined by differentiating (21)-(23) w.r.t. $\gamma$ and reusing (21)-(23):

\begin{align}
r(z') &= -\gamma^{-1}r(z) \tag{24} \\
s' + \gamma \mu^{(0)} \nabla^2 F(x)x' &= -\gamma^{-1}s \tag{25} \\
\tau' \kappa + \kappa' \tau &= -\gamma^{-1} \tau \kappa. \tag{26}
\end{align}
Our algorithm computes $z'$ by solving the linear system of equations (24)-(26). We then compute $\hat{z} = z + \alpha z'$ where $\alpha$ is determined via a line search method such that $\hat{x} \in K$, $\hat{s} \in K^*$ and $\hat{\tau}, \hat{\kappa} \geq 0$ and such that $\hat{z}$ is not too far away from the central path (see Section 5.1.1). We then let $\gamma = \mu(\hat{z})/\mu(z^{(0)})$ and compute a new primal central point $x^+$ by solving (21)-(23) approximately using Newton's method, with $\hat{x}$ as the starting point. After that, we use $x^+$ to determine a new primal-dual feasible point $z^+$ and then repeat. Algorithm 1 shows this entire procedure in detail.

### Algorithm 1 Basic homogeneous algorithm

**Input:** Initial point $z^{(0)} = (x^{(0)}, \tau^{(0)}; y^{(0)}; s^{(0)}; \kappa^{(0)})$ and barrier function $F$.

```
for $k = 0, 1, 2, \ldots$ do
    Residuals: Compute $r(z^{(k)})$ and $r_A^{(k)}$
    Stopping: If stopping criteria satisfied: break
    ATD: Solve the following system for $d^{(k)} = (d_x^{(k)}, d_\tau^{(k)}, d_y^{(k)}, d_s^{(k)}, d_\kappa^{(k)})$:
    
    $r(d^{(k)}) = -r(z^{(k)})$
    $d_\tau^{(k)} + d_\kappa^{(k)} = -\tau^{(k)}\kappa^{(k)}$
    $\mu\nabla^2 F(x^{(k)}) d_x^{(k)} + d_s^{(k)} = -s^{(k)}$

    Line search: Compute a step size $\alpha^{(k)}$
    Update: $\hat{z}^{(k)} := z^{(k)} + \alpha^{(k)} d^{(k)}$ and $\mu := \mu(\hat{z}^{(k)})$ and $\hat{r}_D^{(k)} = r_D(\hat{z}^{(k)})$
    Centering phase:
    Set $w_c^{(0)} = (x_c^{(0)}; \tau_c^{(0)}; y_c^{(0)}; \kappa_c^{(0)}) := (\hat{x}^{(k)}; \hat{\tau}^{(k)}; \hat{y}^{(k)}; \hat{\kappa}^{(k)})$, $\beta^{(0)} = \infty$ and $\lambda^{(0)} = \infty$
    for $j = 0, 1, 2, \ldots$ do
        Residuals: Compute $r_1^{(j)} = r_c^{(j)} - \mu$ and $r_2^{(j)} = r_C(w_c^{(j)}) - z^{(k)}$
        Compute $B = [\|r_1^{(j)}\| \leq \rho_1] \land [\|r_2^{(j)}\| \leq \rho_2] \land [\lambda^{(j)} \leq \rho_3] \land [\beta^{(j)} = 1]$
        CDIR: Solve the following system for $\delta^{(j)} = (\delta_x^{(j)}; \delta_\tau^{(j)}; \delta_y^{(j)}; \delta_\kappa^{(j)})$
        
        $(r_1^{(j)}; r_D(\delta^{(j)})) = (0; 0)$
        $\delta_x^{(j)} + \delta_\kappa^{(j)} r_c^{(j)} = -r_1^{(j)}$
        $\mu\nabla^2 F(x_c^{(j)}) \delta_x^{(j)} - A^T \delta_y^{(j)} + c \delta_\kappa^{(j)} = -r_2^{(j)}$

        Line search: Compute a step size $\beta^{(j+1)}$
        Update: $w_c^{(j+1)} := w_c^{(j)} + \beta^{(j+1)} \delta^{(j)}$ and $\lambda^{(j+1)} = \|\delta_x^{(j)}\|_{\lambda_j^{(j)}}$
    end for
    Primal-dual lifting: Set $s^{(k+1)} := -A^T y_c^{(N)} + c r_c^{(N)} - \hat{r}_D^{(k)}$
    Update: $z^{(k+1)} := (x_c^{(N)}; \tau_c^{(N)}; y_c^{(N)}; s^{(k+1)}; \kappa_c^{(N)})$
end for
```

### 5 Analysis of Algorithm 1

#### 5.1 Approximate Tangent Direction

Let us first list some properties of the approximate tangent direction defined in (27)-(29). For ease of notation in this section, we are going write $\psi$ for $\psi(z^{(k)}, \mu) = s^{(k)} - \mu\nabla^2 F(x^{(k)}) x^{(k)}$ and drop the superscript iteration counter.

**Lemma 2.** Assume $z$ and $d$ satisfy (27)-(29). Then

(i) $d_x^T s + x^T d_s + x^T s = d_x^T \psi$

(ii) $(x + d_x)^T (s + d_s) + (\tau + d_\tau)(\kappa + d_\kappa) = 0$

(iii) $d_x^T d_s + d_x d_\kappa = -d_x^T \psi$. 

(33) (34) (35)
Proof. (i): We get \( d^T s + x^T d_s + x^T s \) \((20)\) \( d^T s + x^T (-s - \mu \nabla^2 F(x)x) + x^T s \), which, after reduction, gives \( d^T s - \mu \nabla^2 F(x)x = d^T \psi \). (ii): Equation (27) is equivalent to \( r(x + d) = 0 \) or \( G(y + d_y; x + d_x; \tau + d_\tau) = 0 \). Pre-multiplying with \((y + d_y; x + d_x; \tau + d_\tau)^T \) gives (34). (iii): Follows from expanding (34) and using (33) and (28).

Lemma 3. Denote \( \hat{z} = z + ad \). Then

\[
\begin{align*}
  r(\hat{z}) &= (1 - \alpha)r(z) \quad (36) \\
  \mu(\hat{z}) &= (1 - \alpha)\mu(z) + \alpha(1 - \alpha)\frac{d^T \psi}{\nu + 1}. \quad (37)
\end{align*}
\]

Proof. Equation (36) follows directly from elementary linear algebra. To show (37):

\[
(\nu + 1)\mu(\hat{z}) = (x + \alpha d_x)^T (s + \alpha d_s) + (\tau + \alpha d_\tau)(\kappa + \alpha d_\kappa) - x^T s + \tau \kappa + \alpha(d^T s + x^T d_s)
\]

\[
+ \alpha(\tau d_\kappa + \kappa d_\tau) + \alpha^2(d^T s + x^T d_s)
\]

\[
\text{Lemma 2}
\]

\[
\begin{align*}
  &\equiv x^T s + \tau \kappa + \alpha(-x^T s + d^T \psi) + \alpha(-\tau \kappa) + \alpha^2(-d^T \psi) \\
  &\equiv (1 - \alpha)(x^T s + \tau \kappa) + \alpha(1 - \alpha)d^T \psi
\end{align*}
\]

which after dividing by \((\nu + 1)\) gives (37).

5.1.1 Line Search

After computing the search direction, we determine a step size \( \alpha \) using backtracking line search [6] to ensure that \( \hat{x} \in \mathcal{K}, \hat{s} \in \mathcal{K}^* \) and \( \tau, \kappa > 0 \). To stay near the central path, we also require

\[
\left\| \left( \frac{\|\nabla F(\hat{x})\|_\infty}{\tau \kappa - \bar{\mu}} \left( \hat{s} + \bar{\mu} \nabla F(\hat{x}) \right) \right) \right\|_\infty \leq \sigma \bar{\mu}, \quad \text{with} \ \sigma \in [0, 1]. \quad (38)
\]

This neighborhood is related to the proximity measure \( \|\mu^{-1} s + \nabla F(x)\|_\infty \). See e.g. [21], Thm. 3.7.1 for a discussion.

5.2 Centering Process

Let \( \hat{\mu} = \mu(\hat{z}^{(k)}) \). After the ATD-step, the goal of the centering process is to find a primal-dual feasible point \( z \) that (approximately) satisfies the equations

\[
\begin{align*}
  r(z) &= r(\hat{z}^{(k)}) \quad (39) \\
  s &= -\hat{\mu} \nabla F(x) \quad (40) \\
  \tau \kappa &= \hat{\mu} \quad (41)
\end{align*}
\]

where \( \hat{\mu} \) is fixed throughout the centering process.

We solve problem (39)-(41) by using Newton’s method. Because of the nonsymmetric nature of this problem, we are not going to require that all iterates stay primal-dual feasible. Instead, we require that all primal iterates be in \( \mathcal{K} \), but not that the dual iterates be in \( \mathcal{K}^* \). For this reason, we eliminate \( s \) from the equations (39)-(41) and get the step equations

\[
\begin{align*}
  A \delta^{(j)} - b \delta^{(j)} &= 0 \quad (42) \\
  -c^T \delta^{(j)} + b^T \delta^{(j)} - \delta^{(j)} &= 0 \quad (43) \\
  r_c^{(j)} \delta^{(j)} + \kappa_c^{(j)} \delta^{(j)} &= \hat{\mu} - r_c^{(j)} \kappa_c^{(j)} \quad (44) \\
  \hat{\mu} \nabla^2 F(x^{(j)}) \delta^{(j)} - A^T \delta^{(j)} + c \delta^{(j)} &= r_D(\hat{z}^{(k)}) - r_C(w^{(j)}). \quad (45)
\end{align*}
\]

We then let \( w^{(j+1)} = w^{(j)} + \beta \delta^{(j)} \), where \( \beta \) is a step length determined by a line search procedure using the merit function

\[
\phi(z) = \|r_D(\hat{z}^{(k)}) - r_C(w_c)\|_\infty + |\tau \kappa - \mu|.
\]
Let $\rho_1$, $\rho_2$ and $\rho_3$ be tolerances. We terminate the process when
\begin{align}
|r_1| &= |\tau_c \kappa_c - \hat{\mu}| \leq \rho_1 \quad (46) \\
\|r_D(\hat{z}(k)) - r_C(w_c)\|_\infty &\leq \rho_2 \quad (47) \\
\|\delta_x\|_{z_c} &\leq \rho_3 \quad (48) \\
\beta &= 1. \quad (49)
\end{align}

**Lemma 4.** The centering problem (39)-(41) can be solved to precision (46)-(49) in polynomial time using Newton’s method with the worst case complexity $O(\phi(\hat{z}))$.

**Proof.** See [9] and [17], Sec. 2.2, Thm 2.2.3 and comments on page 27.

Notice that $\phi(\hat{z})$ is bounded by the neighborhood requirement (38), thus putting an upper bound on the computational effort needed to solve the centering problem.

### 5.3 Primal-dual lifting

After having solved the centering problem to the desired accuracy, we compute a dual point $s^{(k+1)}$ and we thus have a new primal-dual point $z^{(k+1)}$:
\begin{align}
s^{(k+1)} &= -A^T y_c^{(N)} + c r_c^{(N)} - \hat{r}^{(k)} \\
z^{(k+1)} &= (x_c^{(N)}; r_c^{(N)}; y_c^{(N)}; s^{(k+1)}; \kappa_c^{(N)}). \quad (50)
\end{align}

**Lemma 5.** The new primal-dual point satisfies:
\[ r(z^{(k+1)}) = r(\hat{z}(k)). \]

**Proof.** Follows from using elementary linear algebra with (42), (43) and (50).

**Lemma 6.** If $\rho_3 \leq \nu + 1$, the new primal-dual point satisfies
\[ \left| \mu(z^{(k+1)}) - \mu(z^{(k)}) \right| \leq \frac{\rho_1}{\nu + 1}. \]

**Proof.** Because of (48) and (49), we have
\[ w_c^{(N)} = w_c^{(N-1)} + \delta^{(N-1)} \]
\[ \|\delta_x^{(N-1)}\|_{z_c^{(N-1)}} \leq \rho_3 \leq \nu + 1. \]

From (45), we also have:
\[ \hat{\mu} \nabla^2 F(x_c^{(N-1)})\delta_x^{(N-1)} - A^T \delta^{(N-1)} + c r^{(N-1)} = -r^{(N-1)} \Rightarrow \]
\[ \hat{\mu} \nabla^2 F(x_c^{(N-1)})(x_c^{(N-1)} - \delta_x^{(N-1)}) = s^{(k+1)}. \quad (51) \]

We then see that
\[ (\nu + 1)\mu(z^{(k+1)}) = \mu \left( \nabla^2 F(x_c^{(N-1)})(x_c^{(N-1)} - \delta_x^{(N-1)}) \right)^T x_c^{(N)} + \tau_c^{(N)} \kappa_c^{(N)} \]
\[ = \hat{\mu} \left( \nabla^2 F(x_c^{(N-1)})(x_c^{(N-1)} - \delta_x^{(N-1)}) \right)^T (x_c^{(N-1)} + \delta_x^{(N-1)}) + \tau_c^{(N)} \kappa_c^{(N)} \]
\[ = (\nu + 1)\hat{\mu} - \hat{\mu}\|\delta_x^{(N-1)}\|_{x_c^{(N-1)}} + \tau_c^{(N)} \kappa_c^{(N)} - \hat{\mu} \]

and hence
\[ \mu(z^{(k+1)}) = \mu(z^{(k)}) \left( 1 - \frac{\|\delta_x^{(N-1)}\|_{x_c^{(N-1)}}}{\nu + 1} \right) + \frac{r^{(N)}}{\nu + 1} \]
\[ \leq \mu(z^{(k)}) + \frac{r^{(N)}}{\nu + 1} \Rightarrow \]
\[ \left| \mu(z^{(k+1)}) - \mu(z^{(k)}) \right| \leq \frac{\rho_1}{\nu + 1}. \]
Lemma 7. The next $\psi$ satisfies

$$\|\psi^{(k+1)}\|_{\infty} \leq \rho_2.$$  

**Proof.**

$$\psi^{(k+1)} = s^{(k+1)} - \hat{\mu} \nabla^2 F(x^{(k+1)}) x^{(k+1)}$$

$$\equiv s^{(k+1)} + \hat{\mu} \nabla F(x^{(k+1)})$$

$$= -A^T g_c^{(N)} + c r_c^{(N)} - r^{(k)}_D + \hat{\mu} \nabla F(x_c^{(N)})$$

$$= r_C(w_c^{(N)}) - r^{(k)}_D \Rightarrow$$

$$\|\psi^{(k)}\|_{\infty} = \|r_C(w_c^{(N)}) - r^{(k)}_D\|_{\infty} \leq \rho_2.$$  

\[\blacksquare\]

Lemma 8. If $\rho_3 \leq 1$ then the new dual point $s^{(k+1)}$ is dual feasible.

**Proof.** By (32) we have

$$\mu^{-1}s^{(k+1)} + \nabla F(x^{(N-1)}) = -\nabla^2 F(x^{(N-1)}) \delta^{(N-1)}$$

and so

$$\left\| \mu^{-1}s^{(k+1)} + \nabla F(x^{(N-1)}) \right\|_{\nabla F(x^{(N-1)})} \leq \left\| \nabla^2 F(x^{(N-1)}) \delta^{(N-1)} \right\|_{\nabla F(x^{(N-1)})} \Rightarrow$$

$$\|\delta^{(N-1)}\|_{x^{(N-1)}} \leq \lambda^{(N)} \leq \rho_3 \leq 1.$$  

We know by (5) that $-\nabla F(x_c^{(N-1)})$ is a dual feasible point. So by (11), $\mu^{-1}s^{(k+1)}$ is dual feasible, which implies that $s^{(k+1)}$ is dual feasible.

\[\blacksquare\]

5.4 Convergence

The lemmas above imply the convergence of Algorithm 1. Assuming that $F$ is sufficiently regular, there exists $\xi > 0$ such that $\xi$ is a lower bound on all step lengths [6], i.e.

$$\forall k : \alpha^{(k)} \geq \xi.$$  

(52)

Using lemma 3 and (52), we see that, in a worst case sense, getting

$$\|r(z^{(k+1)})\| \leq \epsilon \|r(z^{(0)})\|$$

would take on the order of $O(\xi^{-1} \log (1/\epsilon))$ iterations for $\epsilon \ll 1$.

Lemma 9. Let $\eta \in (0, 1)$ and assume that we in all iterations can obtain

$$(\bar{q}_x^{(k)})^T \psi^{(k)} \leq (\nu + 1)\mu(z^{(k)})$$

(53)

then

$$\mu(z^{(k+1)}) \leq (1 - \eta \xi^2)^k \mu(z^{(0)}).$$

**Proof.** If $\mu(z^{k+1}) \leq \mu(\hat{z}^{(k)})$, then by lemma 3 and (53)

$$\mu(z^{(k+1)}) \leq (1 - \alpha^{(k)})(1 + \alpha^{(k)}) \mu(z^{(k)}).$$

If, on the other hand, $\mu(z^{k+1}) > \mu(\hat{z}^{(k)})$, then by lemma 6, we get

$$(\nu + 1)\mu(z^{(k+1)}) \leq (\nu + 1)\mu(\hat{z}^{(k)}) + \rho_1$$

$$\leq (\nu + 1)(1 - \alpha^{(k)})(1 + \alpha^{(k)}) \mu(z^{(k)}) + \rho_1.$$

8
So by choosing \( \rho_1 = (1 - \eta)(\nu + 1)\mu(z^{(k)})^2 \), we get either way
\[
(\nu + 1)\mu(z^{(k+1)}) \leq (\nu + 1)(1 - (\alpha^{(k)})^2 + (1 - \eta)(\alpha^{(k)})^2)\mu(z^{(k)}) + \rho_1
\]
\[
\leq (\nu + 1)(1 - \eta\xi^2)\mu(z^{(k)})
\]
\[
\leq (\nu + 1)(1 - \eta\xi^2)^k\mu(z^{(0)})
\]
and hence \( \mu(z^{(k+1)}) \leq (1 - \eta\xi^2)^k \mu(z^{(0)}) \).

So to reach the user specified tolerance \( \epsilon \ll 1 \), we require
\[
k \geq \frac{\log \epsilon}{\log (1 - \eta\xi^2)} = O \left( \xi^{-2} \log(1/\epsilon) \right).
\]

**Remark 1.** We ensure that the assumption (53) holds in the following way: After computing an ATD step, we check the inequality (53) and if it does not hold, we take more centering steps thus decreasing \( \|\psi\|_\infty \) (see lemma 7). Eventually (53) will hold. In fact, all of our computational experience with this algorithm shows that (53) *always* holds as long as \( s^{(k)} \) is dual feasible, which is ensured by lemma 8. In practice, this safeguard was therefore never activated. It is possible that (53) can be shown to hold generally under certain assumptions.

**Remark 2.** In [16], a slightly different new primal-dual point \( z^{(k+1)} \) is used. The next primal point is instead \( x^{(k+1)} = x_c^{(N-1)} - \delta_x^{(N-1)} \) while \( \nabla^2 F \) is evaluated at \( x_c^{(N-1)} \). This way, the *exact* scaling relation (51) holds which would imply \( \psi^{(k+1)} = 0 \). With this choice, we would obtain a much simpler right hand side in (37) of lemma 3. However, because the homogenous self-dual model ties the primal and dual problems together via the variables \( \tau \) and \( \kappa \), it would not be possible to choose a combination of \( y^{(k+1)} \), \( \tau^{(k+1)} \) and \( \kappa^{(k+1)} \) in such a way that e.g. lemma 5 would still hold. Further, it certainly makes intuitive sense that it is more beneficial to use the better primaly centered \( x_c^{(N)} \) as the next primal point. Our computational experience indicates that this is better in practice.

**Remark 3.** Although the two lemmas 3 and 9 indicate that \( \mu \) and the residuals \( r_P, r_G \) and \( r_D \) decrease to zero at different rates, we observe in numerical experiments that \( d^{T}_P \psi \) is of much smaller magnitude than \( (\nu + 1)\mu \). Thus we always see \( \mu^{(k+1)} \approx (1 - \alpha)\mu^{(k)} \) in practice.

## 6 Heuristic improvements

In this section we introduce and motivate two heuristic methods to speed up convergence.

### 6.1 Higher order ATD using BFGS updating

Since each iteration of the centering process is as costly as computing a single ATD, it may be worth investing more computational effort in obtaining a search direction that incorporates higher order information. This could lead to a greater decrease in \( \mu \) thus saving many future centering steps.

Let us denote the central path by \( z(\mu) \). The ATD defined in (27)-(29) can be considered an approximation to the tangent to the central path at \( \mu \), i.e. \( z'(\mu) \). Let us write the system (27)-(29) as
\[
K(z,\mu)z'(\mu) = \phi(z,\mu) \quad \text{or} \quad z'(\mu) = K(z,\mu)^{-1}\phi(z,\mu) =: f(z,\mu).
\]

The central path is thus the solution of the ordinary differential equation defined by \( z'(\mu) = f(z,\mu) \). A step in the approximate tangent direction is then the same as taking one Euler step for this ODE. We can, however, obtain a direction that contains, for example, second order information by computing a stage-2 Runge-Kutta direction \( d_2 \), remembering that each evaluation of \( f \) requires solving a system of the type (27)-(29). Such a direction is defined by
\[
d_2 = h \left( 1 - \frac{1}{2\theta} \right) f(z,\mu) + h \frac{1}{2\theta} f(\zeta,\mu(\zeta)) \tag{54}
\]
where \( \zeta = z(\mu) + \theta h f(z,\mu) \tag{55} \)

and where \( h \) is the stepsize possible in the direction \( f(z,\mu) \) and \( \theta \in (0,1) \) is a parameter. The choices \( \theta = 1/2 \) and \( \theta = 1 \) correspond to the classical midpoint and trapezoidal rules respectively [7].
Our experience shows that this approach reduces the total number of iterations as well as the number of factorizations needed for convergence, even though two factorizations are needed to compute $d_2$. We can, however, restrict ourselves to just one factorization by using in place of $\nabla^2 F(\zeta)$ the BFGS update [20] of $\nabla^2 F(x)$. This requires only the extra evaluation of $\nabla F(\zeta)$ and extra back-substitutions. In the next section, we show how this can be done without destroying sparsity in the KKT-system.

### 6.2 BFGS updating in the centering process

Solving either for a centering step or an ATD step requires the factorization of the sparse $n \times n$ matrix $H := \nabla^2 F(x)$ and of the possibly sparse $m \times m$ matrix $Q = AH^{-1}A^T$. To reduce the total number of factorizations needed in the centering process, we suggest taking $J$ BFGS steps for each normal centering step.

Let us show how this can be done computationally efficient. Let $B$ and $M$ denote the BFGS approximation of the inverses of $H$ and $Q$ respectively. Conceptually, we update $B$ to $B^+$ using BFGS updating, a rank-2 updating scheme: 

$$B^+ = B + k^{(v)}v v^T + k^{(w)}w w^T.$$ 

In order to keep the ability to exploit sparsity of $A$ and $Q$, we do not actually store $B$ or $M$ but simply the Cholesky factors of the most recent $H$ and $Q$ and the sequence of BFGS update vectors. More specifically, for $q \leq J$, let $B^{(q)}$ be the $q$’th update of $H^{-1}$, i.e.

$$B^{(q)} = C^{-1}C^{-T} + \Psi \Lambda \Psi^T$$

where $\Psi = [v^{(1)}, \ldots, v^{(q)}, w^{(1)}, \ldots, w^{(q)}], \Lambda = \text{diag}(k_1^{(v)}, \ldots, k_q^{(v)}, k_1^{(w)}, \ldots, k_q^{(w)})$. Then we compute products such as $B^{(q)}r$ by means of

$$B^{(q)}r = C^{-1}(C^{-T}r) + \Psi \left(\Lambda(\Psi^Tr)\right).$$

For $M$, the situation is similar:

$$M^{(q)} = \left(AB^{(q)}A^T\right)^{-1} = \left(A(H^{-1} + \Psi \Lambda \Psi^T)A^T\right)^{-1} = (Q + \Phi \Phi^T)^{-1}$$

where $\Phi = A \Psi$. By the Sherman-Morrison-Woodbury formula, we get

$$M^{(q)} = Q^{-1} - Q^{-1} \Phi (\Lambda^{-1} + \Phi^TQ^{-1}\Phi)^{-1} \Phi^TQ^{-1}.$$ 

We can thus compute products like $M^{(q)}r$ by

$$M^{(q)}r = Q^{-1} \left(I - \Phi (\Lambda^{-1} + \Phi^TQ^{-1}\Phi)^{-1} \Phi^TQ^{-1}\right)r = D^{-1}D^{-T} \left(r - \Phi (\Lambda^{-1} + \Phi^TD^{-1}D^{-T}\Phi)^{-1} \Phi^TD^{-1}D^{-T}r\right)$$

where we remark that 1) only two columns are added to $\Phi$ in each iteration so that only two new backsubstitutions in the product $D^{-1}D^{-T}\Phi$ need to be made, 2) $\Lambda$ is diagonal and thus cheap to invert and 3) the matrix $(\Lambda^{-1} + \Phi^TD^{-1}D^{-T}\Phi)$ is only of size $2q \times 2q$ and is therefore also cheap to invert.

Using this approach, we need to address how to terminate the centering process. An obvious way is to allow termination only after a normal centering step (that is, not a BFGS step), when the criteria (46)-(49) are satisfied. This, however, means that each centering process will involve at least one full centering step.

Alternatively we can, after each BFGS step, check if $\|r_C - \hat{r}_D\|_{*\nabla F(x)} \leq \hat{\mu}_G$. If so, we know that the next dual point is dual feasible (see (50) and lemma 8). Notice that because of (7), the norm $\|v\|_{*\nabla F(x)}$ can be computed as $(v^T H^{-1} v)^{1/2}$. Computing this number requires the evaluation and factorization of $H$. However, since $H$ is blockdiagonal, this operation is cheap. In fact, it is possible simply to analytically compute $H^{-1}$ at each $x$, since $H$ is block diagonal with block sizes $3 \times 3$.

We finally remark that whether or not it is beneficial to take BFGS steps, and if it is, how many should be taken, depends on the cost of building and Cholesky factorizing $AH^{-1}A^T$ relative to the cost of subsequent backsubstitutions. This ratio depends on the dimension and sparsity pattern of $A$, so we can do not say anything about it before knowing $A$. Since the dimension and sparsity pattern of $AH^{-1}A^T$ are the same regardless of the stage of the algorithm, it is possible to determine the above mentioned ratio at initialization time and that way determine how many BFGS steps should be taken in each iteration of the centering process throughout the algorithm.
6.3 Modified algorithm

Our improved algorithm, which is shown in Algorithm 2, implements the pseudo higher order direction described in Section 6.1 and in addition, it takes \( J \) BFGS steps for each normal step, stopping after a BFGS step if \( \| r_C - \tilde{r}_D \|_{F(x)} \leq \tilde{\mu} \rho_3 \) because this seems most effective on the problems we consider. However, we remark that this may not be the case for all problems.

7 Computational results

In this section we present results from running Algorithms 1 and 2 on different test problems. We first introduce the nonsymmetric cones needed for our test problems and then present the test problems. Finally, we include tables with numerical results and discussion.

7.1 Two three-dimensional nonsymmetric cones

In the rest of this paper, we are going to consider problems involving the following two nonsymmetric convex cones.

The three-dimensional exponential cone is defined by

\[
K_{\text{exp}} = \text{closure} \{(x_1; x_2; x_3) \in \mathbb{R} \times \mathbb{R}_+ \times \mathbb{R}_+: \exp(x_1/x_3) \leq x_2/x_3\}
\]

for which we are using the LHSCB

\[
F_{\text{exp}}(x) = -\log (x_3 \log (x_2/x_3) - x_1) - \log x_2 - \log x_3
\]

with barrier parameter \( \nu = 3 \).

The three-dimensional power cone is defined by

\[
K_{\alpha} = \{(x_1; x_2; x_3) \in \mathbb{R} \times \mathbb{R}_+^2 : |x_1| \leq x_2^a x_3^{1-a}\}
\]

where \( \alpha \in [0, 1] \) is a parameter. Notice that \( K_{1/2} \) is the rotated quadratic cone. For all other \( \alpha \in (0, 1) \), \( K_{\alpha} \) is not symmetric. In [8], it was proved that the function

\[
F_{\alpha}(x) = -\log (x_2^{2\alpha} x_3^{2-2\alpha} - x_1^2) - (1 - \alpha) \log x_2 - \alpha \log x_3
\]

is a LHSCB with parameter \( \nu = 3 \) for \( K_{\alpha} \). It is this barrier function we are using in our experiments. Nesterov proposed in [16] a barrier function for the three-dimensional power cone with parameter \( \nu = 4 \). Our computational experiences show that \( F_{\alpha} \) is better in practice.

7.2 Test problems

In this section, \( e \) will denote the vector of all ones. The dimension of \( e \) will be clear from the context.
7.2.1 The $p$-cone problem

Given $A \in \mathbb{R}^{M \times N}$ and $b \in \mathbb{R}^M$, the $p$-cone problem is the problem

$$\min_{x} \|x\|_p, \quad \text{s.t.} \quad Ax = b.$$ 

In [15], it is shown that this is equivalent to

$$\min_{x,y,t} t \quad \text{s.t.} \quad Ax = b, \quad e^T y = t \quad \text{for each} \quad 1 \leq j \leq M.$$ 

7.2.2 The facility location problem

Given $M$ points (locations) in $\mathbb{R}^N$: $C(j)$, $j = 1, \ldots, M$, we want to find the point $z$ with the minimal sum of (weighted) distances to the locations $C(j)$ measured in $p_j$-norms, $p_j \geq 1$. That is

$$\min z \quad \sum_{j=1}^{M} a_j z - C(j) \|_{p_j}$$

where $a_j \geq 0$ are the weights. Let us define the variable

$$x = (z^+; z^-; v^{(1)}; \ldots; v^{(M)}; w^{(1)}; \ldots; w^{(M)}; u^{(1)}; \ldots; u^{(M)}) \in \mathbb{R}^{2N+3MN}.$$ 

We can then formulate (56) as

$$\min_{z^+,z^-,v,w,u} \sum_{j=1}^{M} a_j (v^{(j)} + z^+ - z^-) \quad \text{s.t.} \quad e^T u^{(j)} = u_1^{(j)}, \quad u_1^{(j)} = u_2^{(j)} = \cdots = u_N^{(j)} \quad \text{for each} \quad 1 \leq j \leq M$$

and it is this formulation we use in Section 7.3.

7.2.3 Entropy maximization

Given $A \in \mathbb{R}^{M \times N}$, $b \in \mathbb{R}^M$ and $d \in \mathbb{R}^N$, the entropy maximization problem is

$$\min_{x} \sum_{j=1}^{N} d_j x_j \log x_j$$

$$\text{s.t.} \quad Ax = b \quad x_j \geq 0, \quad j = 1, \ldots, N$$

which can be formulated as

$$\min_{x,u} -d^T u$$

$$\text{s.t.} \quad Ax = b, \quad v = c$$

$$(u_j; v_j; x_j) \in K_{\text{exp}}, \quad j = 1, \ldots, N.$$
7.2.4 Geometric programming

This is a problem of the type

\[
\begin{align*}
\min_{x} & \quad f^{(0)}(x) \\
\text{s.t.} & & g^{(j)}(x) = 1, \quad j = 1, \ldots, M \\
& & f^{(j)}(x) \leq 1, \quad j = 1, \ldots, P
\end{align*}
\]

where \( g^{(j)} \) are monomials and \( f^{(j)} \) are posynomials:

\[
\begin{align*}
g(x) &= k_j x^{b_j}, \\
f^{(j)}(x) &= \sum_{i=1}^{N_j} d_i x^{a_j(i)}
\end{align*}
\]

where we have used the notation

\[
x^v := \prod_{i=1}^{n} x_i^{v_i}, \quad x_i > 0.
\]

With the \( j \)'th posynomial \( f^{(j)} \), we associate

- the matrix \( A^{(j)} := \left( a_1^{(j)}, a_2^{(j)}, \ldots, a_{N_j}^{(j)} \right)^T \in \mathbb{R}^{N_j \times N} \),
- the vector \( d^{(j)} = (d_1^{(j)}, \ldots, d_{N_j}^{(j)})^T \in \mathbb{R}^{N_j \times 1} \) and
- the vector \( c^{(j)} = \log(d^{(j)}) = (\log(d_1), \ldots, \log(d_{N_j}))^T \in \mathbb{R}^{N_j \times 1} \)

Similarly, we associate with the \( j \)'th monomial \( g^{(j)} \)

- the vector \( b^{(j)} \)
- the scalar \( k^{(j)} \)
- the scalar \( h^{(j)} = \log(k^{(j)}) \)

Using the change of variables

\[
u_i = \log(x_i) \quad \iff \quad x_i = \exp(u_i)
\]

for all \( i \), we can write the problem in conic form:

\[
\begin{align*}
\min_{u_+, u_-, w, v, t^{(0)}} & \quad t^{(0)} \\
\text{s.t.:} & & B(u_+ - u_-) + h = 0 \\
& & w^{(j)} = A^{(j)}(u_+ - u_-) + c^{(j)} \quad j = 0, \ldots, P \\
& & e^T v^{(j)} = t^{(j)} \quad y^{(j)} = e \quad j = 0, \ldots, P \\
& & u_+, u_-, t^{(0)} \geq 0 \\
& & \left(w_i^{(j)}; v_i^{(j)}; y_i^{(j)} \right) \in K_{\exp} \quad j = 0, \ldots, P, \ i = 1, \ldots, N_j
\end{align*}
\]

where \( h = (h^{(1)}, \ldots, h^{(M)})^T \in \mathbb{R}^{M \times 1} \) and \( B = (b^{(1)}, \ldots, b^{(M)})^T \in \mathbb{R}^{M \times N} \).

<table>
<thead>
<tr>
<th>Param.</th>
<th>( J )</th>
<th>( \theta )</th>
<th>( \sigma )</th>
<th>( \rho_F )</th>
<th>( \rho_D )</th>
<th>( \rho_A )</th>
<th>( \rho_G )</th>
<th>( \rho_I )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>3</td>
<td>0.70</td>
<td>0.80</td>
<td>10^{-6}</td>
<td>10^{-6}</td>
<td>10^{-6}</td>
<td>10^{-6}</td>
<td>10^{-8}</td>
</tr>
</tbody>
</table>

Table 1: Parameters used.
7.3 Results

Whenever we find a point $z^* = (x^*; \tau^*; y^*; s^*; \kappa^*)$ such that

$$
\| r_P(z^*) \|_\infty \leq \rho_P \land \| r_D(z^*) \|_\infty \leq \rho_D
$$

we conclude that

- if $\| r_A(z^*) \|_\infty \leq \rho_A$ then $(x^*, y^*, s^*)/\tau^*$ is feasible and optimal for (1).
  Otherwise
- if $\| r_G(z^*) \|_\infty \leq \rho_G$ and $\tau^* \leq \rho_I \max(1, \kappa^*)$, (1) is deemed infeasible, specifically
  - primal infeasible if $b^T y^* > 0$
  - dual infeasible if $c^T x^* < 0$.

Throughout we used the parameters displayed in table 1 on the preceding page.

The remaining tables in this section show the number of iteration (it), the total number of factorizations made (chols) and the average number of full centering steps per iteration (ce). Finally, for all but the facility location problems, we show the termination status (st). opt means that an optimal solution was found, ipr/idu means a primal/dual infeasibility certificate was found and roa/roc means that the algorithm was terminated because rounding errors prevented further progress.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Alg. 1</th>
<th>Alg. 2</th>
<th>CVX/SeDuMi</th>
</tr>
</thead>
<tbody>
<tr>
<td>name &amp; size</td>
<td>$p$</td>
<td>it</td>
<td>chol</td>
</tr>
<tr>
<td>stocf01</td>
<td>1.0</td>
<td>11</td>
<td>37</td>
</tr>
<tr>
<td>$M = 117$</td>
<td>3.0</td>
<td>14</td>
<td>62</td>
</tr>
<tr>
<td>$N = 165$</td>
<td>7.0</td>
<td>28</td>
<td>134</td>
</tr>
<tr>
<td>nnz($A$) = 501</td>
<td>12.0</td>
<td>29</td>
<td>137</td>
</tr>
<tr>
<td>20.0</td>
<td>28</td>
<td>131</td>
<td>3.7</td>
</tr>
<tr>
<td>blend</td>
<td>1.0</td>
<td>11</td>
<td>35</td>
</tr>
<tr>
<td>$M = 74$</td>
<td>3.0</td>
<td>13</td>
<td>58</td>
</tr>
<tr>
<td>$N = 114$</td>
<td>7.0</td>
<td>17</td>
<td>77</td>
</tr>
<tr>
<td>nnz($A$) = 522</td>
<td>12.0</td>
<td>19</td>
<td>90</td>
</tr>
<tr>
<td>20.0</td>
<td>19</td>
<td>97</td>
<td>4.1</td>
</tr>
<tr>
<td>share2b</td>
<td>1.0</td>
<td>11</td>
<td>40</td>
</tr>
<tr>
<td>$M = 96$</td>
<td>3.0</td>
<td>12</td>
<td>56</td>
</tr>
<tr>
<td>$N = 162$</td>
<td>7.0</td>
<td>13</td>
<td>62</td>
</tr>
<tr>
<td>nnz($A$) = 777</td>
<td>12.0</td>
<td>13</td>
<td>62</td>
</tr>
<tr>
<td>20.0</td>
<td>13</td>
<td>62</td>
<td>3.8</td>
</tr>
<tr>
<td>share1b</td>
<td>1.0</td>
<td>15</td>
<td>50</td>
</tr>
<tr>
<td>$M = 117$</td>
<td>3.0</td>
<td>17</td>
<td>72</td>
</tr>
<tr>
<td>$N = 253$</td>
<td>7.0</td>
<td>16</td>
<td>67</td>
</tr>
<tr>
<td>nnz($A$) = 1179</td>
<td>12.0</td>
<td>16</td>
<td>66</td>
</tr>
<tr>
<td>20.0</td>
<td>15</td>
<td>64</td>
<td>3.3</td>
</tr>
<tr>
<td>bore3d</td>
<td>1.0</td>
<td>11</td>
<td>25</td>
</tr>
<tr>
<td>$M = 231$</td>
<td>3.0</td>
<td>11</td>
<td>25</td>
</tr>
<tr>
<td>$N = 334$</td>
<td>7.0</td>
<td>11</td>
<td>25</td>
</tr>
<tr>
<td>nnz($A$) = 1444</td>
<td>12.0</td>
<td>12</td>
<td>27</td>
</tr>
<tr>
<td>20.0</td>
<td>12</td>
<td>27</td>
<td>1.2</td>
</tr>
<tr>
<td>scagr25</td>
<td>1.0</td>
<td>14</td>
<td>50</td>
</tr>
<tr>
<td>$M = 471$</td>
<td>3.0</td>
<td>10</td>
<td>37</td>
</tr>
<tr>
<td>$N = 671$</td>
<td>7.0</td>
<td>11</td>
<td>41</td>
</tr>
<tr>
<td>nnz($A$) = 1725</td>
<td>12.0</td>
<td>11</td>
<td>41</td>
</tr>
<tr>
<td>20.0</td>
<td>11</td>
<td>41</td>
<td>2.7</td>
</tr>
<tr>
<td>octap1</td>
<td>1.0</td>
<td>13</td>
<td>43</td>
</tr>
<tr>
<td>$M = 300$</td>
<td>3.0</td>
<td>12</td>
<td>48</td>
</tr>
<tr>
<td>$N = 660$</td>
<td>7.0</td>
<td>14</td>
<td>59</td>
</tr>
<tr>
<td>nnz($A$) = 1872</td>
<td>12.0</td>
<td>18</td>
<td>80</td>
</tr>
<tr>
<td>20.0</td>
<td>21</td>
<td>94</td>
<td>3.5</td>
</tr>
<tr>
<td>bands</td>
<td>1.0</td>
<td>12</td>
<td>42</td>
</tr>
<tr>
<td>$M = 305$</td>
<td>3.0</td>
<td>25</td>
<td>117</td>
</tr>
<tr>
<td>$N = 472$</td>
<td>7.0</td>
<td>31</td>
<td>147</td>
</tr>
<tr>
<td>nnz($A$) = 2494</td>
<td>12.0</td>
<td>31</td>
<td>152</td>
</tr>
<tr>
<td>20.0</td>
<td>30</td>
<td>148</td>
<td>3.9</td>
</tr>
</tbody>
</table>

Table 2: Results for $p$-cone problems.
Table 3: Results for facility location problems. The algorithms always terminated after reaching optimality as all problem instances were feasible by construction.

Table 2 shows results for the $p$-cone problem. We show the results for the two variants of our algorithm and for the solver SeDuMi (see [22]) when called through CVX (see [10]). CVX approximates the solution by solving an approximately equivalent self-scaled problem using SeDuMi.

The matrices used are from the netlib collection. We see that particularly Algorithm 2 performs very well compared to SeDuMi. While the table shows the size of $A$, it does not show the problem size of the final problem solved by either of the solvers. We remark that the number of variables and constraints for Algorithms 1 and 2 stay the same regardless of $p$ while the problem size for SeDuMi grows with $p$. Of course, sufficient sparsity in the problem may mean that it is not necessarily slower. For Algorithm 2, the number of iterations never exceeded 23 and the number of Cholesky factorizations of $AH^{-1}A^T$ never exceeded 43.

Table 3 shows results for facility location problems. The algorithms always terminated after reaching optimality as all problem instances were feasible by construction.

Table 4 shows results from solving a set of real-world entropy problems kindly supplied to us by Erling D. Andersen and Joachim Dahl of MOSEK ApS, Denmark. Generally the problems have many variables compared to the number of constraints resulting in a very “fat” constraint matrix $A$. For these problems we compare our algorithms to the commercial solver from MOSEK, which solves the monotone complementarity problem corresponding to the entropy problem. We see that, except for a few of the problems, Algorithm 1 is clearly inferior to MOSEK’s solver. Our Algorithm 2 performs much better than Algorithm 1, but still used cumulatively about 12% more iterations and about 55% more Cholesky factorizations than MOSEK.

Finally, table 5 shows results from applying our algorithms to a set of geometric programs supplied to us by MOSEK. The column labelled dod denotes the degree of difficulty of the problem. For these problems we see that Algorithm 2 performs quite well producing results comparable to those of MOSEK. Algorithm 2 uses cumulatively about 29% less iterations and 14% less Cholesky factorizations than MOSEK. Notice that Algorithm 2 performs particularly well on the large problems cx02-100, cx02-200, mra01 and mra02, all of which have dod $> 800$. 

<table>
<thead>
<tr>
<th>Problem</th>
<th>Alg. 1</th>
<th>Alg. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>$M$</td>
<td>$\nu$</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>44</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>88</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>128</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>132</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>256</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>128</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>132</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>256</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>128</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>132</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>256</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>128</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>132</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>256</td>
</tr>
<tr>
<td>Problem</td>
<td>Alg. 1</td>
<td>Alg. 2</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>M</td>
</tr>
<tr>
<td>prob</td>
<td>17</td>
<td>15</td>
</tr>
<tr>
<td>prob2</td>
<td>18</td>
<td>14</td>
</tr>
<tr>
<td>ento6</td>
<td>130</td>
<td>21</td>
</tr>
<tr>
<td>ento7</td>
<td>255</td>
<td>21</td>
</tr>
<tr>
<td>ento28</td>
<td>740</td>
<td>16</td>
</tr>
<tr>
<td>ento30</td>
<td>740</td>
<td>21</td>
</tr>
<tr>
<td>ento31</td>
<td>740</td>
<td>21</td>
</tr>
<tr>
<td>ento22</td>
<td>794</td>
<td>28</td>
</tr>
<tr>
<td>ento21</td>
<td>931</td>
<td>28</td>
</tr>
<tr>
<td>a_tb</td>
<td>1127</td>
<td>25</td>
</tr>
<tr>
<td>ento23</td>
<td>1563</td>
<td>28</td>
</tr>
<tr>
<td>ento29</td>
<td>1886</td>
<td>28</td>
</tr>
<tr>
<td>a_12</td>
<td>2183</td>
<td>37</td>
</tr>
<tr>
<td>ento12</td>
<td>2183</td>
<td>37</td>
</tr>
<tr>
<td>a_13</td>
<td>3120</td>
<td>37</td>
</tr>
<tr>
<td>a_23</td>
<td>3301</td>
<td>37</td>
</tr>
<tr>
<td>a_34</td>
<td>3905</td>
<td>37</td>
</tr>
<tr>
<td>a_14</td>
<td>3986</td>
<td>37</td>
</tr>
<tr>
<td>a_35</td>
<td>4333</td>
<td>37</td>
</tr>
<tr>
<td>a_bd</td>
<td>4695</td>
<td>26</td>
</tr>
<tr>
<td>ento2</td>
<td>4695</td>
<td>26</td>
</tr>
<tr>
<td>a_24</td>
<td>5162</td>
<td>37</td>
</tr>
<tr>
<td>ento3</td>
<td>5172</td>
<td>28</td>
</tr>
<tr>
<td>ento50</td>
<td>5172</td>
<td>28</td>
</tr>
<tr>
<td>a_15</td>
<td>5668</td>
<td>37</td>
</tr>
<tr>
<td>a_25</td>
<td>6196</td>
<td>37</td>
</tr>
<tr>
<td>a_36</td>
<td>7497</td>
<td>37</td>
</tr>
<tr>
<td>a_45</td>
<td>7667</td>
<td>37</td>
</tr>
<tr>
<td>ento26</td>
<td>7915</td>
<td>28</td>
</tr>
<tr>
<td>a_16</td>
<td>8528</td>
<td>37</td>
</tr>
<tr>
<td>a_26</td>
<td>9035</td>
<td>37</td>
</tr>
<tr>
<td>ento45</td>
<td>9108</td>
<td>37</td>
</tr>
<tr>
<td>a_46</td>
<td>9455</td>
<td>37</td>
</tr>
<tr>
<td>a_56</td>
<td>9702</td>
<td>37</td>
</tr>
<tr>
<td>ento25</td>
<td>10142</td>
<td>28</td>
</tr>
<tr>
<td>ento41</td>
<td>12691</td>
<td>40</td>
</tr>
<tr>
<td>ento46</td>
<td>15364</td>
<td>31</td>
</tr>
</tbody>
</table>

Table 4: Results for entropy problems.

8 Conclusions and Future Work

We have presented a homogeneous primal-dual interior point algorithm for nonsymmetric conic optimization based on an algorithm previously presented by Nesterov [16]. We have proven convergence of the simplest variant of the algorithm and shown that the efficiency can be significantly improved by employing quasi-Newton techniques to update the Hessian of the barrier function. We have shown how this can be done without losing the ability to exploit sparsity. Finally we have presented extensive computational results that indicate the algorithm works well in practice.

By inspecting the tables in Section 7.3, we see that

- The performance of the algorithms depends a lot on the type of problem.
- Throughout, Algorithm 2 is superior to Algorithm 1.
- For the $p$-cone problems, Algorithm 2 is comparable in performance to SeDuMi when called via CVX.
- For the facility location problems, Algorithm 2 is superior to previously published results [8] of an algorithm similar to the one presented in [16].
- For the entropy maximization problems, Algorithm 1 is clearly inferior to MOSEK while Algorithm 2 is just slightly inferior.
- For the geometric programs, Algorithm 2 performed slightly better than MOSEK when measuring iterations or the total number of Cholesky factorizations.
Table 5: Results for geometric programs.

Comparing the kind of algorithm we have presented with a PDIPM for self-scaled cones, we see that the major difference is the need for a separate centering process. Nesterov remarks in [16] that this process can be seen as the process of finding a scaling point, which naturally is a more complex problem when the cone is not symmetric. We cannot compute it analytically, so an iterative procedure is necessary.

This difference is interesting theoretically as well as practically. For the problems we have considered, the centering problem certainly is a relatively "easy" problem compared to the full problem, in the sense that we do not need a very accurately centered point. We have seen in our experiments with Algorithm 1 that rarely more than 3 or 4 full centering steps are required.

If some of the work in the centering process can be saved – as we have tried to do in Algorithm 2 by employing BFGS updating – it may be possible to bring the total computational effort per iteration close to that of a PDIPM for self-scaled cones. Saving even more work in the centering process is a topic of interest for future work.

Acknowledgments

The authors thank Erling D. Andersen and Joachim Dahl of MOSEK ApS for lots of insights and for supplying us with test problems for the geometric programs and the entropy problems.

References


