CORE DISCUSSION PAPER
2007/71

An interior-point method for the single-facility location problem with mixed norms using a conic formulation

Robert CHARES[1] and François GLINEUR[2]

September 2007

Abstract

We consider the single-facility location problem with mixed norms, i.e. the problem of minimizing the sum of the distances from a point to a set of fixed points in $\mathbb{R}^n$, where each distance can be measured according to a different p-norm.

We show how this problem can be expressed into a structured conic format by decomposing the nonlinear components of the objective into a series of constraints involving three-dimensional cones.

Using the availability of a self-concordant barrier for these cones, we present a polynomial-time algorithm (a long-step path-following interior-point scheme) to solve the problem up to a given accuracy.

Finally, we report computational results for this algorithm and compare with standard nonlinear optimization solvers applied to this problem.

Keywords: nonsymmetric conic optimization, conic reformulation, convex optimization, sum of norm minimization, single-facility location problems, interior-point methods.

[1]CORE and INMA, Université catholique de Louvain, E-mail: robert.chares@uclouvain.be.

[2]CORE and INMA, Université catholique de Louvain, E-mail: francois.glineur@uclouvain.be.

1 Introduction

The single-facility location problem with mixed norms belongs to the class of convex optimization problems, i.e. problems of the form

$$\begin{gathered}\min f(x)\\ x \in S \subseteq \mathbb{R}^n,\end{gathered}$$

where S is a convex set and f is a convex function defined on S. Convex optimization problems have certain advantages over general nonlinear problems, such as the fact that local minima are always also global minima, a rich duality theory and - most importantly - efficient iterative methods, called interior-point methods, able to solve a large class of convex problems in polynomial time.

These methods were initiated by the work of Karmarkar [13] for the linear programming problem. Nesterov and Nemirovski introduced later the notion of self-concordancy [18] in order to derive polynomial-time interior-point methods for convex optimization. A strictly convex function Φ is called a ν-self-concordant barrier for a closed convex set S if it is a barrier for S (i.e. $\Phi : \operatorname{int} S \to \mathbb{R}$ and $\Phi(x) \to \infty$, as $x \to \partial S$) and, defining the local norm of a vector v based on the Hessian of Φ at a given point x as $||v||_x = (v^T \Phi''(x) v)^{1/2}$, if the following three conditions hold (see e.g. [20])

- for all $x \in \operatorname{int} S$ it holds $\{y : ||y-x||_x \le 1\} \subseteq S$,
- for all $x \in \operatorname{int} S$ and $y \in \operatorname{int} S$ such that $||y-x||_x < 1$ it holds
$$1 - ||y-x||_x \le \frac{||v||_y}{||v||_x} \le \frac{1}{1-||y-x||_x}, \quad \forall v \neq 0.$$
- for all $x \in \operatorname{int} S$ it holds $||\nabla \Phi(x)||_x^2 \le \nu$

Constant ν is known as the self-concordance parameter of the barrier and measures its efficiency: indeed, the best interior-point schemes based on the use of such a barrier can be shown to require $\mathcal{O}(\sqrt{\nu}\,\log(\frac{1}{\epsilon}))$ iterations to obtain a solution with ϵ accuracy.

The first author acknowledges support by the Fonds Spéciaux pour la Recherche (FSR) and Fonds pour la Formation à la Recherche dans l'Industrie et dans l'Agriculture (FRIA). This text presents research results of the Belgian Program on Interuniversity Poles of Attraction initiated by the Belgian State, Prime Minister's Office, Science Policy Programming. The scientific responsibility is assumed by the authors.

An important class of convex problems are those in conic format,

$$\begin{aligned} &\min c^T x \\ &s.t.\ Ax = b \\ &\quad x \in K, \end{aligned} \tag{1}$$

for some closed convex cone K. Convex programs in conic form have a strong similarity to linear programs, which correspond to the case $K = \mathbb{R}^n_+$. This similarity can also be observed when writing down their Lagrangean dual

$$\begin{aligned} &\max b^T y \\ &s.t.\ A^T y + s = c \\ &\quad s \in K^*, \end{aligned} \tag{2}$$

where K^* is the dual cone of K, defined by

$$K^* = \{z \mid x^T z \geq 0\ \forall x \in K\}\ .$$

Interior-point schemes based on self-concordant barriers were initially defined for the primal problem alone (or for the dual problem alone), while primal-dual methods were available for linear programming. In 1997, Nesterov and Todd [19] extended the concept of primal-dual algorithms to convex problems in conic form, provided the cone used is assumed to be symmetric, i.e. self-dual and homogeneous. The most important elementary symmetric cones are the nonnegative orthant (leading to linear programming), the second-order cone (leading to convex quadratic programming or second-order cone optimization) and the cone of positive semidefinite matrices (leading to semidefinite programming). However, in the case of non-symmetric cones, no truly primal-dual algorithms (i.e. handling the primal and the dual problem simultaneously and in a completely symmetric fashion) have been proposed yet.

2 The location problem with mixed p-norms

2.1 Related previous work on location problems

Wieszfeld [21] presented in 1937 a simple algorithm to solve the single-facility problem, which started a surge of research on this area. More recently, interior-point methods have been applied to several types of facility location problems. Single-facility location problems based on Euclidean norms were first considered, see for example and Andersen et al. [1], although no complexity result were given. Xue and Ye [22] describe an interior-point method with polynomial complexity to solve single-facility problems using Euclidean norms (cast in conic form), and later in [23] generalize their approach to problems involving arbitrary p-norms with ≥ 1 (mixing different norms was not considered). Den Hertog et al. [6] analyze the complexity of the closely related problem of minimizing a sum of p-powers via interior-point methods.

(Nonconvex) multifacility location problems with forbidden regions are considered in [8], where several new facilities are introduced due to the constraint that they may not be placed at a predefined area. In [7] the authors use the more general concept of gauges to measure the distance between the points. This approach has the advantage that the distance does not need to be symmetric and therefore it might reflect better real-world situation, e.g. movement along or against the current of a river. Carrizosa and Fliege [4] discuss multifacility location problems and their connection to goal programming: in essence, a continuous multifacility location problem can be seen as a particular instance of the non-preemptive goal programming problem, where the deviation to a given target set is to be minimized.

This paper deals with the complexity of an interior-point applied to a conic formulation of a convex optimization problem. Location problems involving multiple facilities being mostly nonconvex (when assignment of the unknown facilities to fixed facilities to be served is not decided beforehand), they fall outside the scope of this paper[1].

2.2 The single-facility location problem with mixed norms

The single-facility location problem with mixed p-norms considered in this paper consists of minimizing the distance of one point $x \in \mathbb{R}^n$ to a certain number of given facilities $B_i \in \mathbb{R}^n, i = 1, \ldots, m$, assuming that the distance to each facility is measured with a p-norm, which is not necessarily the standard Euclidean 2-norm (the p-norm of a vector v is defined as $||v||_p = (\sum_i |v_i|^p)^{1/p}$).

A mathematical formulation of the above situation gives the following unconstrained convex optimization problem

$$\min_{x \in \mathbb{R}^n} g(x) = \min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} c_i ||x - B_i||_{p_i}, \tag{3}$$

[1]See also the remark about multifacility location problems in the concluding Section 6.

where $p_i \geq 1$ and $c_i > 0$ are positive weights for each facility $B_i, i = 1, \ldots, m$. Note that the objective above is convex but non-differentiable when x is equal to one of the facilities B_i, and that p-norms with different values of p can be mixed in the sum. The special case where $p_i = 2$ for all i can be solved using second-order cone programming, see for example [2], Chapter 3. and [22]. We also observe than the number of facilities m can be assumed without loss of generality to be greater or equal to dimension n minus one. Indeed, if we had $m < n - 1$, we could immediately restrict the solution to the affine subspace spanned by the m facilities, which is of dimension $m+1$, and therefore convert the problem into a smaller one where dimension n would become equal to $m + 1$).

Several applications can be modelled as location problems with specific norms that measure the distance between given and new points, such as in transportation [14] or electronic circuit design where cells are placed into a circuit and pairwise connected with wires [5]. The use of mixed norms could prove useful in problems where different transportation modes need to be considered at once (for example, travel distances using air transportation correspond naturally to Euclidean norms, while distances within cities featuring a quasi-rectangular network of streets call for the use of 1-norms, and other types of terrestrial travel time could be modelled with p-norms using exponents between 1 and 2).

We do not explicitly allow side constraints for our location problem. The principal objective of this paper is not to model real-world applications, but merely to show the applicability of our decomposition approach to handle non-Euclidean norm terms in convex problems. From an algorithmic point of view, the inclusion of additional linear and norm constraints would be an easy modification to our model and, while complicating the presentation of our results, would not really influence the design of our interior-point method or its complexity analysis, and is therefore omitted here.

We now proceed to express problem 3 in a structured conic format. Problem (3) is equivalent to the following conic problem involving mixed p-norms

$$\begin{aligned} &\min \sum_{i=1}^{m} c_i t_i \\ &\text{s.t. } ||x - B_i||_{p_i} \leq t_i, \ i = 1, \ldots, m \end{aligned} \tag{4}$$

Indeed, introducing the following p-cone, denoted by P_p, as the epigraph of the corresponding p-norm

$$P_p = \{(x,t) : ||x||_p \leq t\} \subset \mathbb{R}^{n+1}$$

for $p \in [1,\infty)$, which allows the constraints to be written as $(x-B_i, t_i) \in P_{p_i}$, we can cast problem (4) in the dual conic form (2) with the convex cone $K^* = P_{p_1} \times \ldots \times P_{p_m}$. To the best of our knowledge, no self-concordant barrier in closed form with a low self-concordance parameter is known for the p-cone at the moment. Xue and Ye considered in their paper [23] the case $p_i = p, \forall i$ and – based on [18], Section 5.1.1 – used in their implementation a self-concordant barrier for the conic hull of the p-unit-ball. This approach results in a self-concordance parameter of roughly $\mathcal{O}(200)$ for each p-cone in K.

In this paper, we use a different approach: in order to solve this problem with self-concordant barriers featuring lower parameters, we further decompose each of the P_p cones into smaller cones. Following the approach presented recently by Nesterov in [17], we model the p-cone using the so-called power cones. More specifically, we use the following equivalence: $(x,t) \in P_p$ iff $\exists\, y \in \mathbb{R}^n_+$ such that

$$\sum_{j=1}^{n} y_j = t, \tag{5}$$

$$(y_j, t, x_j) \in K_\alpha,\ j = 1, \ldots, n, \tag{6}$$

where $\alpha = 1/p$ and the power cone K_α is defined as

$$K_\alpha = \left\{(z_1, z_2, z_3) \in \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R} \mid z_1^\alpha \cdot z_2^{1-\alpha} \geq |z_3|\right\}. \tag{7}$$

Indeed, it is readily seen that the inequalities (6) imply

$$\sum_{j=1}^{n} |x_j|^p \leq \sum_{j=1}^{n} y_j\, t^{(p-1)} \tag{8}$$

which combined with (5) imply

$$\sum_{j=1}^{n} |x_j|^p \leq t^p.$$

The reverse implication is also obvious (choose each y_j such that (8) is satisfied with equality). In other words, a point (x,t) belongs to the p-cone

if and only if there is a point $y \in \mathbb{R}^n_+$ such that (x, t, y) belongs to a direct product of power cones intersected with a linear subspace.
Using this reformulation for the m p-cones in (4) and substituting t_i in the objective and in the conic constraints, we get the following conic program in the dual format (2) involving a total of $n \cdot m$ power cones

$$
\begin{aligned}
&\min \sum_{i=1}^{m} c_i \sum_{k=1}^{n} y_{ik} \\
&y_{ij}^{\alpha_i} \cdot \left(\sum_{k=1}^{n} y_{ik} \right)^{1-\alpha_i} \geq |x_j - B_{i,j}|,\ i = 1, \ldots, m,\ j = 1, \ldots, n,
\end{aligned}
\tag{9}
$$

or, equivalently,

$$
\begin{aligned}
&\min \sum_{i=1}^{m} c_i \sum_{j=1}^{n} y_{ij} \\
&\left(y_{ij}, \left(\sum_{k=1}^{n} y_{ik} \right), x_j - B_{ij} \right) \in K_{\alpha_i},\ i = 1, \ldots, m,\ j = 1, \ldots, n,
\end{aligned}
\tag{10}
$$

with the cone $K(\alpha) = (K_{\alpha_1} \times \ldots \times K_{\alpha_1}) \times \ldots \times (K_{\alpha_m} \times \ldots \times K_{\alpha_m}) \subset \mathbb{R}^{3nm}$.

We observe that while the original problem (3) contained only n variables, this reformulation involves $n + mn$ variables, the increase being due to the additional y_{ij} variables needed for each of the mn power cones.

Note that the reformulation procedure we have just described is in line with one of the general principles behind the conic formulation: all the nonlinearities of a given (convex) optimization problem should be confined within the cones, while the linear subspace constraints $Ax = b$ should capture the linear part of the problem. In this specific case, each non-linearity of the problem has been broken down to a cone with the smallest possible dimension: one three-dimensional cone for every power function $|\cdot|^p$ present in the original objective $g(x)$ in (3).

3 A path-following algorithm to solve the location problem with mixed norms

The cone $K(\alpha)$ in the previous section does not appear to be symmetric (except when $p = 2$), and there is no obvious way to model or reformulate the cone $K(\alpha)$ as a product of elementary symmetric cones. Therefore primal-dual methods based on symmetric cones as proposed in [19] are not applicable. In this section, we present a standard primal path-following method to solve the location problem with mixed norms.

3.1 A path-following scheme

For a given convex optimization problem

$$\min_{x \in S} c^T x, \tag{11}$$

(where the objective is assumed to be linear without loss of generality), the iterates of path-following interior-point methods follow the central path more or less closely towards an optimal solution. The central path is defined as a family of points $\{x(\mu)\}$, depending on a parameter $\mu > 0$, such that each $x(\mu)$ is the minimizer of the following centering problem

$$\min_{x \in \mathbb{R}} f_\mu(x) = \min_{x \in \mathbb{R}} \frac{1}{\mu} c^T x + F(x), \tag{12}$$

where F is a ν-self-concordant barrier for the feasible set S.

Nesterov and Nemirovski ([18]) showed that the central path is well-defined for all $\mu > 0$ and that indeed $\lim_{\mu \searrow 0} x(\mu) = x^*$, where x^* is an optimal solution of (11). The basic idea of path-following methods is to solve (12) approximately for a decreasing sequence of μ and thus follow the central path to an optimal solution.

In this subsection we will present in more details a path-following scheme designed to solve problem (10), assuming that a ν-self-concordant barrier for the feasible set of (10) is available.

The Newton step at x for the minimization of the objective $f_\mu(x)$ in the unconstrained centering problem (12) is defined as

$$n_\mu(x) = -\nabla^2 f_\mu(x)^{-1} \cdot \nabla f_\mu(x) = -\nabla^2 F(x)^{-1} \cdot \left(\frac{1}{\mu} c + \nabla F(x) \right),$$

and the Newton decrement, which will be used as a proximity measure to the central path, is defined as

$$\begin{aligned} \delta_\mu(x) = ||n_\mu(x)||_x &= \left(n_\mu(x)^T \cdot \nabla^2 f_\mu(x) \cdot n_\mu(x) \right)^{1/2} \\ &= \left(\nabla f_\mu(x)^T \cdot \nabla^2 f_\mu(x)^{-1} \cdot \nabla f_\mu(x) \right)^{1/2} \\ &= \left(\frac{1}{\mu} c + \nabla F(x) \right)^T \nabla^2 F(x)^{-1} \left(\frac{1}{\mu} c + \nabla F(x) \right). \end{aligned}$$

The path-following method described below is a long-step scheme (also called large-update scheme, meaning that multiplying factor θ used to decrease μ at each outer iteration does not depend on the problem dimensions n and m) and uses the following parameters

- $\epsilon > 0$ (desired accuracy on the objective function)
- $0 < \epsilon_c < \frac{1}{4}$ (proximity to the central path)
- $0 < \theta < 1$ (decreasing factor for μ)

and requires a feasible starting point $x^{(0)}$ and the corresponding centering parameter μ_0.

```
k:=0 (outer iteration counter)
i:=0 (inner iteration counter)
while ($\nu \cdot \mu_k > \epsilon \cdot (1 - \epsilon_c)$)
    1. while ($\delta_{\mu_k}(x^{(i)}) > \epsilon_c$)
        (a) compute Newton direction $n_{\mu_k}(x^{(i)})$
        (b) do a (damped) Newton step: $x^{(i+1)} := x^{(i)} + \tau \cdot n_{\mu_k}(x^{(i)})$, where $\tau$ is a suitable step length
        (c) update $i := i + 1$
       end while
    2. update $\mu_{k+1} := \theta \cdot \mu_k$
    3. update $k := k + 1$
end while
```

For suitable choices of the parameters and the starting point, it is possible to prove convergence to an optimal solution in polynomial time, as will be done in Section 4.

If we want to apply the above presented interior-point scheme to solve the location problem (3), we need to have available a self-concordant barrier for the feasible set of the reformulated problem (10).

3.2 A self-concordant barrier for the conic formulation (10)

In the previous subsection we assumed knowledge of a self-concordant barrier for the feasible set of (10). The complexity of interior-point methods depends essentially on the self-concordance parameter of the barrier. Therefore it is advantageous to have a parameter as low as possible.

The following 4-self-concordant barrier for the power cone K_α can be found in [17]:

$$F_\alpha(z_1, z_2, z_3) = -\log(z_1^{2\alpha} z_2^{2(1-\alpha)} - z_3^2) - \log z_1 - \log z_2.$$

From [16, Section 3], using the fact that any affine transformation of a self-concordant barrier remains self-concordant with an unchanged parameter and the fact that the sum of barriers for several sets is a self-concordant barrier for the product of these sets (with a global parameter equal to the sum of the individual parameters), it follows that the following function $F(x, y)$ is a $4nm$-self-concordant barrier for the feasible set of (10):

$$\begin{aligned} F(x, y) =& \sum_{i=1}^{m} \sum_{j=1}^{n} F_{\alpha_i} \left(y_{ij}, \sum_{k=1}^{n} y_{ik}, x_j - B_{ij} \right) \\ =& - \sum_{i=1}^{m} \sum_{j=1}^{n} \log \left((y_{ij})^{2\alpha_i} \cdot \left(\sum_{k=1}^{n} y_{ik} \right)^{2(1-\alpha_i)} - (x_j - B_{ij})^2 \right) \\ & - \sum_{i=1}^{m} \sum_{j=1}^{n} \log y_{ij} - n \sum_{i=1}^{m} \log \sum_{k=1}^{n} y_{ik}. \end{aligned} \tag{13}$$

Computing the gradient and Hessian of this barrier will be required to compute Newton directions. This could be done directly from equation (13) or, more elegantly, by expressing $F(x, y)$ as

$$F(x, y) = \tilde{F}(u, t, v) \tag{14}$$

with

$$\tilde{F}(u, t, v) = \sum_{i=1}^{m} \sum_{j=1}^{n} F_{\alpha_i} (u_{ij}, t_i, v_{ij})$$

and (u, t, v) are defined as the following affine transformation of (x, y),

$$\begin{aligned} u_{ij} &= y_{ij}, \\ t_i &= \sum_{k=1}^{n} y_{ik}, \\ v_{ij} &= x_j - B_{ij}, \end{aligned}$$

which can be written more compactly as

$$\begin{bmatrix} u \\ t \\ v \end{bmatrix} = J \cdot \begin{bmatrix} x \\ y \end{bmatrix} - b \tag{15}$$

where J is a $3mn \times (n + mn)$ matrix and b a vector with $3mn$ components. Using (14) and (15), we have

$$\nabla F(x,y) = J^T \cdot \nabla \tilde{F}\,(u,t,v)\,, \tag{16}$$

$$\nabla^2 F(x,y) = J^T \cdot \nabla \tilde{F}\,(u,t,v) \cdot J, \tag{17}$$

where the gradient and Hessian of $\tilde{F}$ can be easily computed, since the $\nabla\tilde{F}$ can be calculated for each 3-dimensional subvector (u_{ij}, t_i, v_{ij}) separately, and similarly $\nabla^2\tilde{F}$ is block-diagonal with nm size-3 blocks.

4 Proof of polynomial complexity for the algorithm

We now proceed to establish the polynomial complexity of our algorithm.

4.1 Estimating the complexity of each iteration

The most computationally expensive part of a path-following algorithm is the computation of each Newton step, as the solution of a linear system, whose complexity we now try to estimate. Since this system involves $\mathcal{O}(mn)$ variables, a naive upper bound on the complexity of each Newton step is $\mathcal{O}(m^3n^3)$ arithmetic operations. However, this estimate ignores the specific structure of the linear system considered here, which can help decrease the complexity.

Indeed, the linear system corresponding to a general Newton step has the form $\nabla^2 f\ \Delta x = r$ for some right-hand side r. Using automatic differentiation, the cost of carrying out a Hessian$\times$vector multiplication is at most ω times the cost of evaluating the function itself, where ω is a constant smaller than 10 (see [12] for more details). In our case, the function evaluations are $\mathcal{O}(nm)$, which is therefore also the complexity of the corresponding Hessian-vector multiplication. We then use the fact that a conjugate gradient method applied to a Newton system with Hessian (17) requires no more than $n(m+1)$ iterations consisting of such Hessian-vector multiplications. Therefore, the complexity of an efficient method to solve a general dense

Newton system based on a $\mathcal{O}(mn)$ barrier is at most $\mathcal{O}(n^2m^2)$ arithmetic operations.

Moreover, the block structure of the Hessian (17) can be further exploited. Applying a Cholesky factorization after a block elimination using the Schur complement, it can be show that $\mathcal{O}(n^3m)$ arithmetic operations suffice to solve the Newton system (see [3], Appendix C.4), which is better than $\mathcal{O}(n^2m^2)$ when $n \le m$.

In order to obtain the complete algorithmic complexity of our algorithm, we now proceed to estimate the iteration complexity of the algorithm presented in Section 3, as applied to problem (10).

4.2 Estimating the number of outer iterations

We estimate the complexity to obtain a solution x whose objective has a guaranteed ϵ absolute accuracy for problem (3), i.e. such that $g(x) < g(x^*) + \epsilon$, by using the long-step path-following method described in the previous section. Using inequality (2.14) in [20],

$$c^T(x - x^*) \le \mu \cdot \nu \cdot (1 + ||x - x(\mu)||_{x(\mu)})\,, \tag{18}$$

we see that the absolute accuracy of a point x can be bounded by the distance to the corresponding point on the central path. Note, that for short-step methods the theoretical bound on the number of iterations is better ($\sqrt{\nu}$ instead of ν in (18)). However, we implemented a long-step method with fixed decrease of μ because it turned out to be superior in practice, and therefore we will continue our analysis for the long-step scheme. Furthermore, according to [15, Theorem 3.3] it holds that if $\delta_\mu(x) \le \epsilon_c$ (which is true at the end of each outer iteration), then

$$||x - x(\mu)||_{x(\mu)} \le \frac{\delta_\mu(x)}{1 - \delta_\mu(x)} \le \frac{\epsilon_c}{1 - \epsilon_c}.$$

That means we have an ϵ-solution if $\delta_\mu(x) \le \epsilon_c$ and

$$\mu \cdot \nu \cdot \left(1 + \frac{\epsilon_c}{1 - \epsilon_c}\right) = \mu \cdot \nu \cdot \left(\frac{1}{1 - \epsilon_c}\right) \le \epsilon$$

$$\Leftrightarrow \qquad \mu \cdot \nu \le \epsilon \cdot (1 - \epsilon_c).$$

Estimating now the number of outer iterations necessary to reach

$$\mu < \frac{1}{\nu}\epsilon \cdot (1 - \epsilon_c),$$

we use the fact that $\mu_k = \mu_0 \theta^k$ so that it suffices to run

$$\frac{\log \epsilon(1-\epsilon_c) - \log \nu \mu_0}{\log \theta}$$

outer iterations to obtain an ϵ-solution, provided our iterates are centered according to $\delta_\mu(x) < \epsilon_c$ after each outer iteration. That means that for our problem, the number of outer iterations can be bounded by

$$\mathcal{O}\left(\log\left(\frac{4mn}{\epsilon}\right)\right).$$

4.3 Estimating the number of centering steps

We now continue the complexity analysis by estimating the number of inner iterations needed for each centering problem. For an initial point $x^{(i)}$ close to the central path, i.e. $\delta_{\mu_k}(x^{(i)}) < \epsilon_c$, the number of centering steps towards the new target point on the central path $x(\mu_{k+1})$ in each outer iteration can be bounded using the self-concordance property of (12) in the following way.

The functional difference to the new target point on the central path

$$f_{\mu_{k+1}}(x^{(i)}) - f_{\mu_{k+1}}(x(\mu_{k+1}))$$

can be bounded (see [20, Section 2.4.3.]) by

$$\frac{1}{\theta} \cdot (\nu + \sqrt{\nu}).$$

Moreover, the functional decrease in each inner iteration by applying damped Newton steps can be guaranteed to be at least a constant as long as $\delta_{\mu_{k+1}}(x^{(i)}) \geq \epsilon_c$. Indeed, if we choose

$$\tau = \frac{1}{\delta_{\mu_{k+1}}(x^{(i)}) + 1} \text{ such that } x^{(i+1)} = x^{(i)} + \frac{n_{\mu_{k+1}}(x^{(i)})}{\delta_{\mu_{k+1}}(x^{(i)}) + 1},$$

it clearly holds that $x^{(i+1)} \in B(x^{(i)}, 1) = \{y : ||y - x^{(i)}||_{x^{(i)}} \leq 1\}$. We can then apply [20, Theorem 2.2.2] and get

$$f_{\mu_{k+1}}(x^{(i)}) - f_{\mu_{k+1}}(x^{(i+1)}) \geq \sigma(\epsilon_c) > 0,$$

where $\sigma(\epsilon_c)$ is an absolute constant depending only on ϵ_c. In practice, of course, other steps lengths may be chosen as long as they decrease the function value by at least $\sigma(\epsilon_c)$.

Finally, this shows that take at most

$$\frac{1}{\sigma(\epsilon_c)\cdot\theta}\left(\nu+\sqrt{\nu}\right)$$

steps before we obtain a point $x^{(i+1)}$ that is again close to the central path for the updated μ, i.e.

$$\delta_{\mu_{k+1}}(x^{(i)}) < \epsilon_c$$

and therefore the number of iterations for each inner loop is bounded by $\mathcal{O}(mn)$.

However, this analysis does not hold for the first inner loop because the initial point $x^{(0)}$ is not necessarily centered (δ_{μ_0} can be greater than ϵ_c).

4.4 Estimating the number of iterations for the initial centering

For this part of the analysis, we cannot keep the generality of the previous sections and have to use some specific information about our problem. Indeed, let us formulate the centering problem (12) for the special case of the conic problem (10),

$$\min_{x\in\mathbb{R}^n} f_\mu(x,y) \tag{19}$$

where

$$f_\mu(x,y)=\frac{1}{\mu}\sum_{i=1}^{m}c_i\sum_{j=1}^{n}y_{ij}+F(x,y)$$

and $F(x,y)$ is defined as in (13).

We consider now the centering problem (19) and in the following assume without loss of generality that $c_i\in[c_{\min},1]$, $\forall i=1,\dots,m$, where $c_{\min}>0$ and $B_{ij}\in[0,1]$, $\forall i,j$. Indeed, these assumptions are not restrictive because by scaling and translating the original data, we can always generate a problem that satisfies these conditions.

As a given starting point $(x^{(0)},y^{(0)})$ does not necessarily satisfy the centering condition that has been imposed in the previous subsection, we have to estimate the number of iterations needed to generate such a point close to the central path, i.e. close to the minimizer $(x(\mu_0),y(\mu_0))$ of the centering problem (19), starting from some initial guess $(x^{(0)},y^{(0)})$.

The functional difference of the objective of the centering problem (19) between $(x^{(0)}, y^{(0)})$ and $(x(\mu_0), y(\mu_0))$ is

$$\begin{aligned} &f_{\mu_0}(x^{(0)}, y^{(0)}) - f_{\mu_0}(x(\mu_0), y(\mu_0)) \\ &\qquad = \frac{1}{\mu_0} \sum_{i=1}^{m} c_i \sum_{j=1}^{n} y_{ij}^{(0)} + F(x^{(0)}, y^{(0)}) - f_{\mu_0}(x(\mu_0), y(\mu_0)). \qquad (20) \end{aligned}$$

Using the same arguments as in the previous subsection, this difference is responsible for the number of iterations for the first inner loop and we proceed now to bound it from above.

We choose the following values for the initial iterate:

- $\mu_0 = 1$
- $x^{(0)} = [1/2, \ldots, 1/2]^T \in \mathbb{R}^n$
- $y^{(0)} = [1, \ldots, 1]^T \in \mathbb{R}^{nm}$

and it can be easily checked that the point $(x^{(0)}, y^{(0)})$ is feasible for the problem (10).

First, we show that $\sum_{i=1}^{m} c_i \sum_{j=1}^{n} y_{ij}^{(0)}$ in (20) can be bounded. In fact,

$$\sum_{i=1}^{m} c_i \sum_{j=1}^{n} y_{ij}^{(0)} = n \cdot \sum_{i=1}^{m} c_i \leq nm.$$

As the next step, we need to bound the initial value of the barrier, i.e. find a $C_1(m, n)$ such that $F(x^{(0)}, y^{(0)}) \leq C_1(m, n)$.

It can be seen that

$$\begin{aligned} F(x^{(0)}, y^{(0)}) = &- \sum_{i=1}^{m} \sum_{j=1}^{n} \left[\log \left(\left(y_{ij}^{(0)} \right)^{2\alpha_i} \cdot \left(\sum_{k=1}^{n} y_{ik}^{(0)} \right)^{2(1-\alpha_i)} - \left(x_j^{(0)} - B_{ij} \right)^2 \right) \right. \\ &\left. + \log y_{ij}^{(0)} + \log \sum_{k=1}^{n} y_{ik}^{(0)} \right] \\ \leq &- \sum_{i=1}^{m} \sum_{j=1}^{n} \left[\log \left(1 - \frac{1}{4} \right) + \log 1 + \log n \right] \\ = &- nm \log \left(\frac{3}{4} n \right) \leq C_1(m, n) = \mathcal{O}(mn \log(n)). \end{aligned}$$

In the last step, we need to bound from below the value of the objective of the initial centering problem (19) in order to estimate the term of (20). For $\mu_0 = 1$ let

$$\begin{aligned} -f_{\mu_0}(x,y) &= -\sum_{i=1}^{m} c_i \sum_{j=1}^{n} y_{ij} - \sum_{i=1}^{m}\sum_{j=1}^{n} F_{\alpha_i}(x,y) \\ &= \sum_{i=1}^{m} h_i(x,y) \end{aligned}$$

where

$$\begin{aligned} h_i(x,y) =& - c_i \sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \log y_{ij} + n \log\left(\sum_{k=1}^{n} y_{ik}\right) \\ &+ \sum_{j=1}^{n} \log\left(y_{ij}^{2\alpha_i} \left(\sum_{k=1}^{n} y_{ik}\right)^{2-2\alpha_i} - (x_j - B_{ij})^2 \right) \\ \leq& - c_i \sum_{j=1}^{n} y_{ij} + (2\alpha_i + 1) \sum_{j=1}^{n} \log\left(y_{ij}\right) + (2 - 2\alpha_i + n) \log\left(\sum_{k=1}^{n} y_{ik}\right). \end{aligned}$$

The maximizer of the right-hand side function has to satisfy the following system of equations,

$$\begin{array}{cccc} c_i & -\frac{2\alpha_i+1}{y_{i1}} & -\frac{2-2\alpha_i+n}{\sum_{j=1}^{n} y_{ij}} & = 0 \\ \vdots & \vdots & \vdots & \vdots \\ c_i & -\frac{2\alpha_i+1}{y_{in}} & -\frac{2-2\alpha_i+n}{\sum_{j=1}^{n} y_{ij}} & = 0 \end{array}$$

from which it follows

$$y_{i1} = \cdots = y_{in} =: \vartheta_i$$

and

$$\begin{aligned} & c_i - \frac{2\alpha_i + 1}{\vartheta_i} - \frac{2 - 2\alpha_i + n}{\vartheta_i n} = 0 \\ \Rightarrow & \vartheta_i = \frac{(2 + 2/n)(1 + \alpha_i)}{c_i}. \end{aligned}$$

Therefore,

$$\begin{aligned}
h_i(x,y) \leq & -c_i n\left(\frac{(2+2/n)(1+\alpha_i)}{c_i}\right)+(2\alpha_i+1)\sum_{j=1}^{n}\log\left(\frac{(2+2/n)(1+\alpha_i)}{c_i}\right)\\
&+(2-2\alpha_i+n)\log\left(n\,\frac{(2+2/n)(1+\alpha_i)}{c_i}\right)\\
=& -(2n+2)(\alpha_i+1)+n(2\alpha_i+1)\log\left(\frac{(2+2/n)(1+\alpha_i)}{c_i}\right)\\
&+(2-2\alpha_i+n)\log\left(\frac{(2n+2)(1+\alpha_i)}{c_i}\right)\\
\leq & -(2n+2)+3n\log(6)-(4n+2)\log(c_{\min})+(n+2)\log(4n+4)\\
=&\mathcal{O}\big(n\log(\frac{n}{c_{\min}})\big)
\end{aligned}$$

for all $i=1,\dots,m$. This implies that $f_{\mu_0}(x,y)$ can be bounded by a constant $C_2(m,n,c_{\min})$

$$-f_{\mu_0}(x,y)\leq C_2(m,n,c_{\min})=\mathcal{O}\big(m\,n\log(\frac{n}{c_{\min}})\big)$$

and, in particular, that this holds for $(x(\mu_0),y(\mu_0))$, the last term in (20). That means the total difference of the objective of the centering problem (19) between $(x^{(0)},y^{(0)})$ and $(x(\mu_0),y(\mu_0))$ is no more than

$$nm-nm\log\left(\frac{3}{4}n\right)+C_2(m,n,c_{\min})=:C(m,n,c_{\min})=\mathcal{O}\big(m\,n\log(\frac{n}{c_{\min}})\big).$$

For the same reasons as in the previous subsection, we can argue that it is possible to reduce the objective value of the initial centering problem (19) in each iteration of a damped Newton method by at least a constant $\sigma(\epsilon_c)$. It follows that in

$$\mathcal{O}\left(\frac{C(m,n,c_{\min})}{\sigma(\epsilon_c)}\right)=\mathcal{O}\big(m\,n\log(\frac{n}{c_{\min}})\big)$$

steps we obtain a point close to the central path.

From a theoretical point of view, an auxiliary path-following scheme to find the analytic center of the feasible set (to provide an initial iterate close to the central path) would feature a better complexity estimate $\mathcal{O}(\sqrt{nm}\log(nm))$ than the scheme we implemented. This would consist in solving an auxiliary problem with another objective vector $\tilde{c}$ such that an arbitrary starting interior point is exactly on the central path. One would then follow the central path to the analytic center of the feasible set using a short-step path-following scheme with the above mentioned complexity.

However, since the rest of the algorithm is using damped long steps and has already iteration complexity $\mathcal{O}(mn)$, our choice of also using a damped Newton scheme for the initialization strategy has no negative impact on the overall complexity of the method. Furthermore, it turned out that these estimates are too conservative in practice and that the damped Newton scheme very often finds a good approximation for a point close to the central path in no more than 5 iterations.

We are ready now to formulate the final theorem.

Theorem 4.1. *Let* $c_i \in [c_{\min}, 1]$, $\forall i = 1, \dots, m$, *where* $c_{\min} > 0$ *and* $B_{ij} \in [0, 1]$, *for* $i = 1, \dots, m, j = 1, \dots, n$. *The long-step path-following method described in Section 3 initialized as described above solves the unconstrained location problem with mixed norms*

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} c_i ||x - B_i||_{p_i},$$

in $\mathcal{O}(n\,m) \cdot \mathcal{O}\left(\log\left(\frac{n\,m}{\epsilon c_{\min}}\right)\right)$ *iterations.*

Proof. Recall that the first centering step needs at most $\mathcal{O}\big(m\,n\,\log(\frac{n}{c_{\min}})\big)$ iterations. The total number of outer iterations is then bounded by $\mathcal{O}\left(\log\left(\frac{m\,n}{\epsilon}\right)\right)$ steps, and the number of centering steps for each outer iteration is bounded by $\mathcal{O}(m\,n)$ steps, so that the total complexity is

$$\begin{aligned} &= \mathcal{O}\big(m\,n\,\log(\frac{n}{c_{\min}})\big) + \mathcal{O}(m\,n) \times \mathcal{O}\left(\log\left(\frac{m\,n}{\epsilon}\right)\right) \\ &= \mathcal{O}(n\,m) \cdot \mathcal{O}\left(\log\left(\frac{n\,m}{\epsilon c_{\min}}\right)\right) \text{ iterations.} \end{aligned}$$

$\square$

5 Computational results

We implemented the long-step path-following scheme described in Section 3 in MATLAB and tested it on some randomly generated problems where the exponents p_i are uniformly distributed on the [1 3] interval and the facilities are uniformly distributed in the $[0\ 1]^n$ box. The positive weights c_i in the objective function are chosen to be all equal to 1.

We choose μ_0 and x_0 according to Section 4.4, the centering accuracy $\epsilon_0 = 10^{-1}$, a long-step update with $\theta = 0.1$ and an absolute objective accuracy of $\epsilon = 10^{-6}$.

5.1 Comparison with nonlinear AMPL solvers

For each choice of problem parameters (n, m) presented here, 10 different instances were solved and the average is reported. We compare our solver with the AMPL nonlinear solvers MINOS[2] , LOQO[3] and KNITRO[4], all used with their default parameters (no attempt was made to tune them specifically to our problem) and limited to 2000 iterations per problem.

MINOS is using a quasi-Newton method to solve nonlinear problems. LOQO uses a primal-dual interior-point method applied to a quadratic approximation of the original problem. KNITRO is based on a direct barrier method to solve a primal-dual KKT system, using trust regions and a merit function are used to promote convergence.

Table 1 reports the number of iterations (major iterations for KNITRO) carried out by each method.

Table 1: Number of iterations for each solver (averaged on 10 instances)

dimension	IPM solver	MINOS	LOQO	KNITRO
$n = 2, m = 10$	27.6	12.9	206.7	22.9
$n = 2, m = 100$	33.3	7.2	57.9	8.8
$n = 2, m = 1000$	43.6	7.7	33.8	8.7
$n = 2, m = 10000$	51.5	7.8	9.1	5.5
$n = 10, m = 10$	42.8	128.9	206.1	29.8
$n = 10, m = 50$	46.8	34	217.6	40.9
$n = 10, m = 100$	55.3	128.4	263.5	21.4
$n = 10, m = 500$	60.4	44.1	301.8	29.5
$n = 10, m = 1000$	59	25.4	203.3	16
$n = 50, m = 10$	53.6	243.6	314.8	215.3
$n = 50, m = 50$	70.1	245.1	453.1	177.6
$n = 50, m = 70$	70.1	171.4	341.4	364.7
$n = 50, m = 100$	67.5	141.2	452.4	89.4
$n = 50, m = 200$	82.3	261.8	457.4	1133.1

[2]http:// www.sbsi-sol-optimize.com/asp/sol_product_minos.htm

[3]http://www.princeton.edu/~rvdb/loqo/

[4]http://www.ziena.com/knitro.htm

The iteration numbers of our interior-point solver are much better than what could be expected from theory, since the theoretically guaranteed $\mathcal{O}(n \cdot m)$ iteration bound seems to be quite pessimistic: in practice, the number of iterations barely triples between $nm = 20$ and $nm = 10000$ (a usual phenomenon for interior-point methods).

Although not competitive for the small value $n = 2$, iteration counts become comparable for $n = 10$ to those of KNITRO, the best nonlinear solver, and are clearly better for $n = 50$. The guidance of the central path is thus clearly beneficial here.

It is interesting to see that when fixing n and increasing the value of m, the number of iterations for all the AMPL nonlinear solvers remains constant or even decreases. This effect could be explained by a "smoothing-out" of the objective function. For a large m, non-differentiable terms in the objective become "small" with respect to the complete sum of the norms, and the objective is almost smooth. This potentially explains why the nonlinear solvers find an optimal solution faster for large values of m, even though the problems seem to be more difficult.

Table 2 shows the computation times in seconds[1]. Please bear in mind that this comparison across solvers is not completely fair since we cannot expect MATLAB, an interpreted language, to be as fast as those native compiled solvers. To somehow support this claim, we report that for a typical $(n, m) = (50, 100)$ run, 50% of the CPU time is spent on building the Hessian (involving a lot of data manipulation within MATLAB) and only 30% on actually computing the Newton step (solving a linear system with a single MATLAB command), while the latter operation should in principle be dominating the CPU cost.

Our interior-point solver is not competitive with the AMPL nonlinear solvers. Only LOQO's computing times could be ranked roughly in the same category as ours, although this is mainly due to LOQO having trouble reaching the required accuracy and stopping because of its iteration limit.

The main explanation for the poor performance of the interior-point scheme seems to be the large number of variables needed for the conic formulation: indeed, instead of working with a vector of n unknowns (and, accordingly, computing a $n \times n$ Hessian), our algorithm requires an additional y_{ij} variable for each of the mn cones involved, for a total of $n(m+1)$ variables and the corresponding much enlarged Hessian.

Table 3 displays the percentage of problems for which the AMPL solvers claimed to have found an optimal solution. The need for this table was

[1]Intel Pentium IV 3.00 GHz; MATLAB version 7.2 (R14)

Table 2: CPU time (in seconds) used by each solver (averaged on 10 instances)

dimension	IPM solver	MINOS	LOQO	KNITRO
$n = 2, m = 10$	0.07	0.10	0.12	0.10
$n = 2, m = 100$	0.31	0.10	0.18	0.11
$n = 2, m = 1000$	4.00	0.16	0.31	0.16
$n = 2, m = 10000$	136.37	0.70	1.13	0.64
$n = 10, m = 10$	0.30	0.17	0.18	0.11
$n = 10, m = 50$	1.38	0.16	0.60	0.16
$n = 10, m = 100$	3.26	0.72	0.98	0.16
$n = 10, m = 500$	19.33	1.00	8.24	0.68
$n = 10, m = 1000$	36.00	0.89	9.15	0.56
$n = 50, m = 10$	4.47	0.51	1.03	0.46
$n = 50, m = 50$	40.38	2.45	6.64	1.38
$n = 50, m = 70$	58.35	1.91	7.51	3.64
$n = 50, m = 100$	80.87	2.18	15.28	1.64
$n = 50, m = 200$	198.50	11.56	43.66	39.14

prompted by that fact that, for a significant number of instances, the AMPL solvers could not satisfy their stopping criterion (based on the norm of the gradient) and stopped either because a built-in iteration limit (LOQO) or insufficient progress after a certain number of iterations (MINOS, KNITRO), reporting that the final iterate *might* not be optimal. For large m and n, this problematic behavior even seems to become the norm. However, in nearly all the cases, the solution provided was indeed optimal (meaning in that case that all six requested digits of accuracy matched between the interior-point solution and its nonlinear counterpart), except in the case of LOQO, where some unsuccessful runs stopped because of the iteration limit exhibited a less accurate solution (meaning a few mismatched digits when comparing to the interior-point and other nonlinear solutions). Nonetheless, we still classify these situations as unsuccessful because in general we do not know how close to the optimal solution we are and would like to have a guarantee to be within an ϵ distance of the optimal solution.

These failures to detect optimality are most probably due to the proximity/equality of the optimal solution to one of the fixed facilities and the (near) non-differentiability of the objective function that it causes. It is remarkable to observe that this non-differentiability has a significant impact on the practical behavior of the AMPL codes tested, even on relatively

Table 3: Percentage of solutions for which optimality is guaranteed

dimension	IPM solver	MINOS	LOQO	KNITRO
$n = 2, m = 10$	100%	70%	60%	70%
$n = 2, m = 100$	100%	90%	90%	90%
$n = 2, m = 1000$	100%	100%	100%	90%
$n = 2, m = 10000$	100%	100%	100%	100%
$n = 10, m = 10$	100%	90%	60%	90%
$n = 10, m = 50$	100%	80%	60%	70%
$n = 10, m = 100$	100%	90%	60%	70%
$n = 10, m = 500$	100%	70%	50%	50%
$n = 10, m = 1000$	100%	100%	90%	90%
$n = 50, m = 10$	100%	80%	40%	40%
$n = 50, m = 50$	100%	60%	10%	10%
$n = 50, m = 70$	100%	60%	40%	10%
$n = 50, m = 100$	100%	90%	10%	10%
$n = 50, m = 200$	100%	40%	10%	0%

simple unconstrained problems with a finite number of problematic points. One can therefore conclude here that one of the main advantages of the interior-point solver lies in its insensitivity to these issues.

5.2 Comparison with Xue and Ye's algorithm

The authors of [23] present an algorithm to solve the similar - but not identical - problem of minimizing a sum of p-norms, where all norms in the objective are defined by one single value of p and the decision variable x is scaled by a matrix A_i^T in each norm term,

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} ||B_i - A_i^T x||_p, \tag{21}$$

with $B_i \in \mathbb{R}^d$ and $A_i^T \in \mathbb{R}^{d \times n}$, $i = 1, \ldots, m$. They propose a nonsymmetric primal-dual potential-reduction method that relies on the self-concordant barrier for the conic hull of the p-unit ball. Due to this construction the self-concordance parameter becomes relatively large, i.e. $200m(2d + 1)$ for the description they chose in the computational examples.

We considered the problem (3)

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} c_i ||x - B_i||_{p_i}$$

which is slightly more general in the sense that it does not require the norm terms to have identical parameters p, but on the other hand does not make use of scaling matrices A_i^T (however, incorporation of these matrices in our model would be trivial).

Although a direct comparison is not possible, problem (21) can be rewritten as

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} ||B_i|| \cdot \left|\left| \frac{B_i}{||B_i||} - \frac{A_i^T}{||B_i||} x \right|\right|_p ,$$

with the constants in each norm term having components in the interval $[0, 1]$. Assuming that $d = n$ and $p_i = p, \forall i = 1, \ldots, m$, we can compare the complexity of both methods in that case. The self-concordance parameter of the barrier used by Xue and Ye in [23] is $200m(2n+1)$, while it is only $4nm$ for our barrier, showing a clear advantage for our approach. The iteration complexity of the method used by Xue and Ye in their computational results is $\mathcal{O}(2m\sqrt{200m(2n+1)} \cdot \log(\frac{\max ||B_i||}{\epsilon} \cdot mn))$, for our method is $\mathcal{O}(n\,m) \cdot \mathcal{O}\left(\log\left(\frac{n\,m}{\epsilon c_{\min}}\right)\right)$, which are equivalent except when $m \gg n$, in which case our method will have a better bound. Finally, the cost per iteration in [23] is $\mathcal{O}(mn^3)$, which is also the case for our method. Summarizing, both methods have comparable overall algorithmic complexity, although our method has a slight advantage when $m \gg n$.

We now look at the test case considered in [23] of finding the shortest network under a given N-Steiner topology, with $N = 10$. We reformulate this problem as a sum of p-norms problem yields the parameters $m = 2N - 3, d = 2, n = 2N - 4$. They consider several values of p, among others $p = 3$. With the algorithms proposed in their paper they get a solution with an accuracy of $1.0e - 5$ in 33 iterations, whereas choosing for our formulation some random data in the same dimensions (omitting d), we get a solution with the same accuracy requirement in 31 iterations. This slight improvement is not too surprising because of the smaller self-concordance parameter of our barrier, although both iteration counts are much better than their corresponding pessimistic worst-case bounds.

Comparing CPU times is not possible since no computation times are reported in [23]. Altough the size of the linear systems to be solved at each iteration is smaller $(2m(d + 1) + 2m + n = 152)$ when compared to ours

($n(m+1) = 288$), the special block-structure and sparsity of our system has to be taken into account, which make it difficult to predict which iteration will be more efficient in practice.

To conclude this section, we observe that the main advantage of our approach seems to be its simplicity and versatility: a single barrier for the 3-dimensional power cone is all that is needed to derive a polynomial-time algorithm, while the approach in [23] proposes three different barriers for p-cones, to be chosen according to the value of the norm parameter p. Moreover, while the approach of Xue and Ye can in principle be applied to any problem involving p-cones such as (4), ours can be applied to any problem based on power cones, which encompass all problems with p-cones and many others (such as problems involving sums of p powers).

6 Concluding remarks

We have showed that to solve the location problem with mixed norms, the proposed approach based on a decomposition into a structured conic format is successful. This approach allows us to generate an extended formulation that can be solved efficiently by an interior-point method with a guaranteed total algorithmic complexity of $\mathcal{O}(n\,m) \cdot \mathcal{O}\left(\log\left(\frac{n\,m}{\epsilon c_{\min}}\right)\right)$ iterations, each involving $\mathcal{O}(n^2m^2)$ arithmetic operations.

Although the location problem (3) considered in this paper is rather simple, we would like to point out the fact that the approach described can be very easily extended to more complicated problems involving side constraints, such as requiring the unknown facility to belong to some polyhedral region, or imposing upper bounds on (the sum of) some distance terms similar to those involved in the objective function. Indeed, any of those problems can easily be reformulated into a conic model similar to (10) involving only power cones (using the fact that both p-norm and affine nonnegativity constraints can be modelled using the power cone), to which our algorithm can be applied without modification. Further classes of problems can be handled in exactly the same fashion, such as convex Euclidean multifacility location problems (EMFL) and Euclidean Steiner minimal tree (SMT) problems considered in [22], and their variants based on p-norms instead of Euclidean norms, since these problems are also reducible to conic models using power cones.

We observed that, in practice, the number of iterations increases only slowly with the dimension, and becomes lower than the number iterations used by the three nonlinear solvers MINOS, LOQO and KNITRO for problems involving 50 unknowns. Unfortunately, this is compensated by the high cost per iteration inherent to our approach, due to the increased number of variables of the conic formulation. Yet, our approach has the advantage of being very reliable with respect to finding a guaranteed optimal solution, and unsensitive to the non-differentiability of the objective function.

The large number of variables present in our formulation has negative implications for the building of the Hessian and the computation of the Newton step. However, it might be possible to take advantage of the structure and sparsity of the Hessian and speed up these computations. In this respect, formulas (16) and (17) and the very particular sparsity pattern of matrix J could prove helpful.

Another possibility to improve this scheme's computational efficiency would be to resort to the notion of *partial minimization* of the self-concordant barrier, which works as follows: instead of keeping a vector of variables y as iterates, it could be possible (with some adaptations) to compute efficiently the minimum of the barrier $F(x, y)$ with respect to the variables y. The interior-point scheme would then only work with vector x as iterate and the resulting partially minimized barrier function depending only on x (along with its gradient and Hessian), which is guaranteed to be self-concordant with the same parameter as the original barrier $F(x, y)$. We leave the exploration of this technique for further research.

Recently, Nesterov [17] presented a primal-dual predictor-corrector method that generates iterates in the primal-dual space by computing the corrector steps in the primal space and generating the predictor step for the primal-dual variables, and for which the dual barrier does not need to be available explicitly. Therefore, this method is applicable for general non-symmetric cones, and in particular to our problem and its conic formulation. Further research could be done to implement it and compare it with our primal-only path-following scheme.

Finally, we would like to emphasize the fact that the general approach we followed here, relying on the decomposition of a given convex optimization problem into a conic problem with many small cones followed by the application of a standard interior-point scheme, can in principle be adapted to many other classes of convex problems, including for example geometric programming problems [9], l_p-norm optimization [11] and convex optimization with separable objective and constraints[10, Chapter 7].

Acknowledgements: The authors would like to thank Yurii Nesterov for fruitful discussions and comments about this paper.

References

[1] Knud D. Andersen, Edmund Christiansen, Andrew R. Conn, and Michael L. Overton. An efficient primal-dual interior-point method for minimizing a sum of euclidean norms. *SIAM Journal on Scientific Computing*, 22(1):243–262, 2000.

[2] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization. Analysis, Algorithms, and Engineering Applications.* SIAM, Philadelphia, 2001.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[4] E. Carrizosa and J. Fliege. Generalized goal programming: Polynomial methods and applications. *Mathematical Programming*, 93(2):281–303, 2002.

[5] M. del Mar Hershenson, S.P. Boyd, and T.H. Lee. Optimal design of a cmos op-amp via geometric programming. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(1):1–21, 2001.

[6] D. den Hertog, F. Jarre, C. Roos, and T. Terlaky. A sufficient condition for self-concordance with application to some classes of structured convex programming problems. *Mathematical Programming*, 69:75–88, 1995.

[7] J. Fliege. Solving convex location problems with gauges in polynomial time. *Studies in Locational Analysis*, 14:153–172, 2000.

[8] J. Fliege and S. Nickel. An interior point method for multifacility location problems with forbidden regions. *Studies in Locational Analysis*, 14:23–46, 2000.

[9] F. Glineur. Proving strong duality for geometric optimization using a conic formulation. *Annals of Operations Research*, 105:155–184, July 2001.

[10] F. Glineur. *Topics in Convex Optimization: Interior-Point methods, Conic Duality and Approximations.* Ph.D. thesis, Faculté Polytechnique de Mons, Mons, Belgium, January 2001.

[11] F. Glineur and T. Terlaky. Conic formulation for l_p-norm optimization. *Journal of Optimization Theory and Applications*, 122(2):285–307, 2004.

[12] A. Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation.* Frontiers in Applied Mathematics, 19, SIAM, Philadelphia, 2000.

[13] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[14] R.F. Love, J.G. Morris, and G.O. Wesolowsky. *Facilities Location: Models & Methods.* North Holland, Amsterdam, 1988.

[15] S. Nash and A. Sofer. On the complexity of a practical interior-point method. *SIAM Journal of optimization*, 8(3):833–849, 1998.

[16] Y. Nesterov. Interior-point methods: An old and new approach to nonlinear programming. *Mathematical Programming*, 79:285–297, 1997.

[17] Y. Nesterov. Towards nonsymmetric conic optimization. *CORE Discussion Paper*, 28, 2006.

[18] Y. Nesterov and A. Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM, Philadelphia, 1994.

[19] Y. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research*, 22:1–42, 1997.

[20] J. Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization.* SIAM, Philadelphia, 2001.

[21] E. Weiszfeld. Sur le point par lequel le somme des distances de n points donnés est minimum. *Tohoku Math. J.*, 4:355–386, 1937.

[22] G. Xue and Y. Ye. An efficient algorithm for minimizing a sum of euclidean norms with applications. *SIAM Journal on Optimization*, 7(4):1017–1036, 1997.

[23] G. Xue and Y. Ye. An efficient algorithm for minimizing a sum of p-norms. *SIAM Journal on Optimization*, 10(2):551–579, 1998.