Available online at www.sciencedirect.com
ScienceDirect
Applied Mathematics and Computation 189 (2007) 1368–1383
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Two unconstrained optimization approaches for the Euclidean κ-centrum location problem

Shaohua Pan [a,*], Jein-Shan Chen [b]

[a] School of Mathematical Sciences, South China University of Technology, Guang zhou, 510640, China
[b] Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan

Abstract

Consider the single-facility Euclidean κ-centrum location problem in $\mathbb{R}^n$. This problem is a generalization of the classical Euclidean 1-median problem and 1-center problem. In this paper, we develop two efficient algorithms that are particularly suitable for problems where n is large by using unconstrained optimization techniques. The first algorithm is based on the neural networks smooth approximation for the plus function and reduces the problem to an unconstrained smooth convex minimization problem. The second algorithm is based on the Fischer–Burmeister merit function for the second-order cone complementarity problem and transforms the KKT system of the second-order cone programming reformulation for the problem into an unconstrained smooth minimization problem. Our computational experiments indicate that both methods are extremely efficient for large problems and the first algorithm is able to solve problems of dimension n up to 10,000 efficiently.

Keywords: The Euclidean κ-centrum problem; Smoothing function; Merit function; Second-order cone programming

1. Introduction

Given a positive integer number κ in the interval $[1, m]$, the single-facility Euclidean κ-centrum problem in $\mathbb{R}^n$ concerns locating a new facility so as to minimize the sum of the κ largest weighted Euclidean distances to the existing m facilities. Let $a_i \in \mathbb{R}^n$ represent the position of the ith existing facility and $x \in \mathbb{R}^n$ denote the unknown position of the new facility. Let

$$f_i(x) = \omega_i \sqrt{(x_1 - a_{i1})^2 + (x_2 - a_{i2})^2 + \cdots + (x_n - a_{in})^2}, \quad i = 1, 2, \ldots, m$$

* Corresponding author.
E-mail addresses: shhpan@scut.edu.cn (S. Pan), jschen@math.ntnu.edu.tw (J.-S. Chen).

0096-3003/$ - see front matter © 2007 Published by Elsevier Inc.
doi:10.1016/j.amc.2006.12.014

be the weighted Euclidean distance between the new facility and the ith existing facility, where $\omega_i > 0$ is the associated weight. Then the problem can be formulated as the following minimization problem:

$$\min_{x\in\mathbb{R}^n} \Phi_\kappa(x) := \sum_{l=1}^{\kappa} f_{[l]}(x), \tag{1}$$

where $f_{[1]}(x), f_{[2]}(x), \ldots, f_{[m]}(x)$ are the functions obtained by sorting $f_1(x), f_2(x), \ldots, f_m(x)$ in nonincreasing order, namely for any $x \in \mathbb{R}^n$,

$$f_{[1]}(x) \geqslant f_{[2]}(x) \geqslant \cdots \geqslant f_{[\kappa]}(x) \geqslant \cdots \geqslant f_{[m]}(x).$$

There are many different kinds of facility location problems and for which there have been proposed various methods; see [9,10,14] and the related literature in the web-site maintained by EWGLA (Euro Working Group on Locational Analysis). The single-facility Euclidean κ-centrum problem studied here is to generalize the classical Euclidean 1-median problem (corresponding to the case $\kappa = m$) and 1-center problem (corresponding to the case $\kappa = 1$) from a view of solution concept. To our knowledge, the κ-centrum concept was first defined by Slater [25] and Andreatta and Mason [3,4] for the discrete single-facility location problem, and later in [23] was extended to general location problems covering discrete and continuous decisions. In addition, Tamir et al. [22,28] recently did some excellent works related to the κ-centrum location problem, especially the rectilinear κ-centrum problem.

The current research for Euclidean facility location problems mainly focuses on the 1-median problem, for which many practical and efficient algorithms have been designed since Weiszfeld presented a simple iterative algorithm in 1937. These include the hyperboloid approximation procedure [11], the interior point algorithms [1,29,30], the smoothing Newton methods [19,20] and the merit function approach [7]. However, the solution methods for the κ-centrum location problem are rarely seen in the literature except [2,21], where the problem of minimizing the κ largest Euclidean norms is only mentioned as a special example of a second-order cone programming and consequently can be solved by an interior point method. The main purpose of this paper is to develop two efficient algorithms for the single-facility Euclidean κ-centrum problem in $\mathbb{R}^n$, which can be used to handle the cases where n is large (say, in the order of thousands).

Note that problem (1) is a nonsmooth convex optimization problem. Due to the nondifferentiability of the objective function, the gradient-based algorithms can not be used to solve the problem directly. To overcome this difficulty, we reduce (1) as a nonsmooth problem only involving the plus function $\max\{0, t\}$, and then utilize the neural network smoothing function [6] to give a convex approximation. Based on the approximation problem, we propose a globally convergent quasi-Newton algorithm. In addition, we reformulate (1) as a standard primal second-order cone programming (SOCP) problem to circumvent its nondifferentiability. This SOCP reformulation is completely different from the ones in [2,21], and particularly we here use a merit function approach rather than an interior point method to solve it. More specifically, we convert its Karush–Kuhn–Tucker (KKT) system into an equivalent unconstrained smooth minimization problem by the Fischer–Burmeister merit function [8] for second-order cone complementarity problems (SOCCPs), and then solve this smooth optimization problem with a limited-memory BFGS method. In contrast to interior point methods, our unconstrained optimization methods do not require an interior point starting, and moreover have the advantages of requiring less work per iteration, thereby rendering themselves to large problems.

The rest of this paper proceeds as follows. In Section 2, we derive a smooth approximation to (1). Based on the smooth approximation, we design a globally convergent quasi-Newton algorithm in Section 3. In Section 4, we present the detailed process of reformulating (1) as a standard primal SOCP problem. Based on its KKT system and the Fischer–Burmeister merit function for SOCCPs, we develop another quasi-Newton algorithm in Section 5. In Section 6, we report our preliminary computational results and compare our algorithms with the SeDuMi 1.05 (a primal-dual interior point algorithm for the SOCP and the semidefinite programming). The results show that our first algorithm is the most effective by CPU time and able to solve problems of dimension n up to 10,000, whereas our second algorithm is comparable with even superior to the SeDuMi for the moderate problems (say, in the order of hundreds). Finally, we conclude this paper in Section 7.

In this paper, unless otherwise stated, all vectors are column vectors. We use I_d to denote the $d \times d$ identity matrix and 0_d to denote a zero vector in $\mathbb{R}^d$. To represent a large matrix with several small matrices, we use

semicolons ";" for column concatenation and commas "," for row concatenation. This notation also applies to vectors. For a convex function $f:\mathbb{R}^n \to \mathbb{R}$, we let $\partial f(x)$ denote the subdifferential of f at x. Note that f is minimized at x over $\mathbb{R}^n$ if and only if $0 \in \partial f(x)$. For the calculus rules on subdifferentials of convex functions, please refer to [16] or [24].

2. Smooth approximation

In this section, we reduce problem (1) to a nonsmooth optimization problem only involving the plus function $\max\{0,t\}$, and then give a smooth approximation via the neural networks smoothing function proposed by [6]. We also demonstrate that the smooth approximation can be generated by regularizing problem (1) with a binary entropy function.

First, it is not hard to verify that the function $\Phi_\kappa(x)$ in (1) can be expressed by

$$\Phi_\kappa(x) = \max\left\{\sum_{i=1}^m \lambda_i f_i(x) : \sum_{i=1}^m \lambda_i = \kappa, 0 \leqslant \lambda_i \leqslant 1, i = 1,2,\ldots,m\right\}, \tag{2}$$

since, on the one hand, for any feasible point $\lambda = (\lambda_1,\ldots,\lambda_m)^{\mathrm{T}}$ of the maximization problem in (2), there always holds

$$\begin{aligned}\sum_{i=1}^m \lambda_i f_i(x) &= \lambda_{i_1} f_{[1]}(x) + \lambda_{i_2} f_{[2]}(x) + \cdots + \lambda_{i_m} f_{[m]}(x)\\ &\leqslant \lambda_{i_1} f_{[1]}(x) + \cdots + \lambda_{i_k} f_{[k]}(x) + \lambda_{i_{k+1}} f_{[k]}(x) + \cdots + \lambda_{i_m} f_{[k]}(x)\\ &\leqslant \lambda_{i_1} f_{[1]}(x) + \cdots + \lambda_{i_k} f_{[k]}(x) + (\kappa - \lambda_{i_1} - \cdots - \lambda_{i_k}) f_{[k]}(x)\\ &\leqslant \Phi_\kappa(x)\end{aligned}$$

on the other hand, there exists an feasible point $\tilde{\lambda} = \left(\tilde{\lambda}_1,\ldots,\tilde{\lambda}_m\right)^T$ such that $\tilde{\lambda} = \left(\tilde{\lambda}_1,\ldots,\tilde{\lambda}_m\right)^T$, where $\tilde{\lambda}_i = 1$ if $f_i(x)$ belongs to the κ largest function in the collection $\{f_i(x)\}_{i=1}^m$, and otherwise $\tilde{\lambda}_i = 0$. We note that the linear programming problem in (2) has the following dual problem:

$$\begin{aligned}\min_{w,\eta}\quad & \kappa w + \sum_{i=1}^m \eta_i\\ \text{s.t.}\quad & \eta_i \geqslant f_i(x) - w, \quad i = 1,2,\ldots,m,\\ & \eta_i \geqslant 0, i = 1,2,\ldots,m,\end{aligned} \tag{3}$$

and moreover, they both have nonempty feasible sets for any $x \in \mathbb{R}^n$ where $\eta = (\eta_1,\cdots,\eta_m)^{\mathrm{T}}$ and w and η_i are the Lagrange multipliers associated with the constraints $\sum_{i=1}^m \lambda_i = \kappa$ and $\lambda_i \leqslant 1$, respectively. From the duality theory of linear programming, $\Phi_\kappa(x)$ can then be represented by the dual problem (3). However, we observe that each pair of constraints $\eta_i \geqslant f_i(x) - w$ and $\eta_i \geqslant 0$ in (3) can be replaced with one constraint $\eta_i \geqslant \max\{f_i(x) - w, 0\}$, whereas the objective function of (3) is increasing componentwise in η. Therefore,

$$\Phi_\kappa(x) = \min_{w\in\mathbb{R}}\left\{\kappa w + \sum_{i=1}^m \max\{0, f_i(x) - w\}\right\}, \tag{4}$$

Thus, problem (1) reduces to a nonsmooth problem only involving the plus function:

$$\min_{w\in\mathbb{R}, x\in\mathbb{R}^n}\left\{\kappa w + \sum_{i=1}^m \max\{0, f_i(x) - w\}\right\}. \tag{5}$$

To the best of our knowledge, the equivalent formulation (5) has not been found in the literature though the derivation procedure above is very simple.

In [6], Chen and Mangasarian presented a class of smooth approximations to the plus function $\max\{0,t\}$. Among these smooth approximations, the neural networks smooth function and the Chen–Harker–Kanzow–Smale smooth function are most commonly used. In this paper we will use the neural networks smoothing function defined by

$$\varphi(t,\varepsilon) = \varepsilon \ln[1 + \exp(t/\varepsilon)] \quad (\varepsilon > 0). \tag{6}$$

By $\varphi(t,\varepsilon)$, we define the function

$$\Phi(w,x,\varepsilon) := \kappa w + \sum_{i=1}^{m} \varphi(f_i(x,\varepsilon) - w, \varepsilon), \tag{7}$$

where

$$f_i(x,\varepsilon) = \omega_i \sqrt{\|x - a_i\|^2 + \varepsilon^2}, \quad i = 1,2,\ldots,m.$$

Then, by Lemma 1 below, we can reformulate problem (1) as an unconstrained smooth convex minimization problem

$$\min_{w\in\mathbb{R}, x\in\mathbb{R}^n} \Phi(w,x,\varepsilon). \tag{8}$$

Lemma 1. *The function $\Phi(w,x,\varepsilon)$ defined as in (7) has the following properties*:

(i) *For any $w \in \mathbb{R}, x \in \mathbb{R}^n$, and $\varepsilon_1, \varepsilon_2$ satisfying $0 < \varepsilon_1 < \varepsilon_2$, we have*

$$\Phi(w,x,\varepsilon_1) < \Phi(w,x,\varepsilon_2); \tag{9}$$

(ii) *For any $x \in \mathbb{R}^n$ and $\varepsilon > 0$,*

$$\Phi_\kappa(x) \leqslant \min_{w\in\mathbb{R}} \Phi(w,x,\varepsilon) \leqslant \Phi_\kappa(x) + m(\ln 2 + 1)\varepsilon; \tag{10}$$

(iii) *For any $\varepsilon > 0$, $\Phi(w,x,\varepsilon)$ is continuously differentiable and strictly convex.*

Proof

(i) For any $t \in \mathbb{R}$, by a simple computation, we have

$$\frac{\partial\varphi(t,\varepsilon)}{\partial\varepsilon} = \ln[1 + \exp(t/\varepsilon)] - \frac{\exp(t/\varepsilon)}{1+\exp(t/\varepsilon)} \cdot \frac{t}{\varepsilon} > 0$$

and

$$\frac{\partial\varphi(t,\varepsilon)}{\partial t} = \frac{\exp(t/\varepsilon)}{1+\exp(t/\varepsilon)} > 0.$$

The two inequalities imply that for any $w \in \mathbb{R}, x \in \mathbb{R}^n$, and $\varepsilon_1, \varepsilon_2$ satisfying $0 < \varepsilon_1 < \varepsilon_2$,

$$\Phi(w,x,\varepsilon_1) < \kappa w + \sum_{i=1}^{m} \varphi(f_i(x,\varepsilon_2) - w, \varepsilon_1) < \kappa w + \sum_{i=1}^{m} \varphi(f_i(x,\varepsilon_2) - w, \varepsilon_2) = \Phi(w,x,\varepsilon_2).$$

(ii) For any $t \in \mathbb{R}$ and $\varepsilon > 0$, it is easy to verify that

$$\max\{0,t\} \leqslant \varphi(t,\varepsilon) \leqslant \max\{0,t\} + \varepsilon \ln 2.$$

This implies that for any $w \in \mathbb{R}, x \in \mathbb{R}^n$ and $\varepsilon > 0$,

$$\max\{0, f_i(x,\varepsilon) - w\} \leqslant \varphi(f_i(x,\varepsilon) - w, \varepsilon) \leqslant \max\{0, f_i(x,\varepsilon) - w\} + \varepsilon \ln 2, \quad i = 1,\ldots,m,$$

whereas

$$\max\{f_i(x) - w, 0\} \leqslant \max\{f_i(x,\varepsilon) - w, 0\} \leqslant \max\{f_i(x) - w, 0\} + \varepsilon, \quad i = 1,2,\ldots,m.$$

The two sides imply that

$$\kappa w + \sum_{i=1}^{m} \max\{0, f_i(x) - w\} \leqslant \Phi(w,x,\varepsilon) \leqslant \kappa w + \sum_{i=1}^{m} \max\{0, f_i(x) - w\} + m(\ln 2 + 1)\varepsilon.$$

From the inequality and (4), the conclusion immediately follows.

(iii) For any $\varepsilon > 0$, clearly, $\Phi(w,x,\varepsilon)$ is continuously differentiable. Now we prove that it is strictly convex. From (7),

$$\nabla\Phi(w,x,\varepsilon) = \begin{pmatrix} \kappa - \sum_{i=1}^{m} \lambda_i(w,x,\varepsilon) \\ \sum_{i=1}^{m} \frac{\lambda_i(w,x,\varepsilon)\omega_i(x-a_i)}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \end{pmatrix}, \tag{11}$$

where

$$\lambda_i(w,x,\varepsilon) = \frac{\exp[(f_i(x,\varepsilon)-w)/\varepsilon]}{1+\exp[(f_i(x,\varepsilon)-w)/\varepsilon]}, \quad i = 1,2,\ldots,m. \tag{12}$$

Let

$$\hat{\lambda}_i(w,x,\varepsilon) = \lambda_i(w,x,\varepsilon) - \lambda_i^2(w,x,\varepsilon), \quad i = 1,2,\ldots,m,$$

and

$$Q = \sum_{i=1}^{m} \left[\frac{\lambda_i(w,x,\varepsilon)\omega_i}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \left(I_n - \frac{(x-a_i)(x-a_i)^{\mathrm{T}}}{\|x-a_i\|^2+\varepsilon^2} \right) + \frac{\hat{\lambda}_i(w,x,\varepsilon)\omega_i^2}{\varepsilon(\|x-a_i\|^2+\varepsilon^2)} (x-a_i)(x-a_i)^{\mathrm{T}} \right].$$

Then, it follows from (11) that

$$\nabla^2\Phi(w,x,\varepsilon) = \begin{pmatrix} \sum_{i=1}^{m} \frac{\hat{\lambda}_i(w,x,\varepsilon)}{\varepsilon} & -\sum_{i=1}^{m} \frac{\hat{\lambda}_i(w,x,\varepsilon)\omega_i(x-a_i)^{\mathrm{T}}}{\varepsilon\sqrt{\|x-a_i\|^2+\varepsilon^2}} \\ -\sum_{i=1}^{m} \frac{\hat{\lambda}_i(w,x,\varepsilon)\omega_i(x-a_i)^{\mathrm{T}}}{\varepsilon\sqrt{\|x-a_i\|^2+\varepsilon^2}} & Q \end{pmatrix}.$$

For any $\varepsilon > 0$ and $z = (z_0; z) \in \mathbb{R}^{n+1}$ with $z \neq 0$, we have

$$\begin{aligned} z^{\mathrm{T}}\nabla^2\Phi(w,x,\varepsilon)z &= \varepsilon^{-1}z_0^2 \sum_{i=1}^{m} \hat{\lambda}_i(w,x,\varepsilon) - 2\varepsilon^{-1}z_0 \sum_{i=1}^{m} \frac{\hat{\lambda}_i(w,x,\varepsilon)}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \omega_i(x-a_i)^{\mathrm{T}}z + z^{\mathrm{T}}Qz \\ &= \varepsilon^{-1} \sum_{i=1}^{m} \hat{\lambda}_i(w,x,\varepsilon) \left[z_0^2 - 2z_0 \frac{\omega_i(x-a_i)^{\mathrm{T}}z}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} + \left(\frac{\omega_i(x-a_i)^{\mathrm{T}}z}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \right)^2 \right] \\ &\quad + \sum_{i=1}^{m} \frac{\lambda_i(w,x,\varepsilon)\omega_i}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \left(\|z\|^2 - \left(\frac{(x-a_i)^{\mathrm{T}}z}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \right)^2 \right) \\ &\geqslant \sum_{i=1}^{m} \frac{\lambda_i(w,x,\varepsilon)\omega_i}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \left[\|z\|^2 - \left(\frac{(x-a_i)^{\mathrm{T}}z}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \right)^2 \right] \\ &\geqslant \sum_{i=1}^{m} \frac{\lambda_i(w,x,\varepsilon)\omega_i}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \left(\|z\|^2 - \|z\|^2 \cdot \left\| \frac{x-a_i}{\sqrt{\|x-a_i\|^2+\varepsilon^2}} \right\|^2 \right) > 0 \end{aligned}$$

and moreover $z^{\mathrm{T}}\nabla^2\Phi(w,x,\varepsilon)z = 0$ if and only if $z = 0$. The first inequality is due to the nonnegativity of $\hat{\lambda}_i(w,x,\varepsilon)$, the second follows from the Cauchy–Schwartz inequality and the nonnegativity of $\lambda_i(w,x,\varepsilon)$, and the last is obvious. Here, the proof of lemma is completed. □

To close this section, we give a different view into the smooth approximation (8). Let $\theta(t) = t\ln t + (1-t)\ln(1-t)$. Introduce the binary entropy function $\sum_{i=1}^{m}\theta(\lambda_i)$ as a regularizing term to the objective of linear programming problem in (2) and construct the regularization problem

$$\max\left\{\sum_{i=1}^{m}\lambda_i f_i(x,\varepsilon) - \varepsilon\sum_{i=1}^{m}\theta(\lambda_i) : \sum_{i=1}^{m}\lambda_i = \kappa,\ 0 \leqslant \lambda_i \leqslant 1,\ i = 1,2,\ldots,m\right\}. \tag{13}$$

Clearly, (13) is a strictly convex programming problem and satisfies the Slater constraint qualification. Hence, from the duality theory of convex program, the optimal value of (13) is equivalent to that of its dual problem. By a simple computation, (13) has the following dual problem

$$\min_{w\in\mathbb{R}} \Phi(w,x,\varepsilon).$$

Compared with the previous Eq. (8), this means that the smooth approximation in (8) is actually equivalent to the following binary entropy regularization problem

$$\min_{x\in\mathbb{R}^n}\max\left\{\sum_{i=1}^{m}\lambda_i f_i(x,\varepsilon) - \varepsilon\sum_{i=1}^{m}\theta(\lambda_i) : \sum_{i=1}^{m}\lambda_i = \kappa,\ 0 \leqslant \lambda_i \leqslant 1,\ i = 1,2,\ldots,m\right\}.$$

We note that Shi [26] constructed a similar regularization problem to derive a smoothing function for the sum of the κ largest components, but he did not provide an explicit expression for his smoothing function so that the function value must be determined by numerical computations.

3. Algorithm based on the neural networks smoothing function

In what follows, we present an algorithm for solving problem (1) based on the smooth approximation (8), followed by a global convergence result.

Algorithm 1. Let $\sigma \in (0,1)$, $(\hat{w}^0,\hat{x}^0) \in \mathbb{R}^{n+1}$ and $\varepsilon_0 > 0$ be given. Set $k := 0$.

For $k = 0,1,2,\ldots,$ do

1. Use an unconstrained optimization method with $(\hat{w}^k,\hat{x}^k)$ as the starting point to solve

$$\min_{w\in\mathbb{R},x\in\mathbb{R}^n} \Phi(w,x,\varepsilon_k), \tag{14}$$

and write its minimizer as (w^k,x^k).

2. Set $\varepsilon_{k+1} = \sigma\varepsilon_k$, $(\hat{w}^k,\hat{x}^k) = (w^{k+1},x^{k+1})$, and then go back to Step 1.

End

Lemma 2. *Let x be any point in $\mathbb{R}^n$ and define the index set*

$$I(x) := \{i \in \{1,2,\ldots,m\} | f_i(x) \geqslant f_{[\kappa]}(x)\}. \tag{15}$$

Then

$$\partial\Phi_\kappa(x) = \left\{\sum_{i\in I(x)}\rho_i\partial f_i(x) : \sum_{i\in I(x)}\rho_i = \kappa, 0 \leqslant \rho_i \leqslant 1 \text{ for } i \in I(x), \rho_i = 0 \text{ for } i \notin I(x)\right\}. \tag{16}$$

Proof. Let $\vartheta_\kappa(y) = \sum_{l=1}^{\kappa} y_{[l]}$, where $y_{[1]},\ldots,y_{[m]}$ are the numbers $y_1,\ldots,y_m$ sorted in nonincreasing order. Then, it follows from (2) that

$$\vartheta_\kappa(y) = \max\left\{\sum_{i=1}^{m}\lambda_i y_i : \sum_{i=1}^{m}\lambda_i = \kappa,\ 0 \leqslant \lambda_i \leqslant 1,\ i = 1,2,\ldots,m\right\} = \delta^*(y|C),$$

where $\delta^*(\cdot\|C)$ denotes the support function of the set C and

$$C = \left\{ \lambda \in \mathbb{R}^m \middle| \sum_{i=1}^{m} \lambda_i = \kappa,\ 0 \leqslant \lambda_i \leqslant 1,\ i = 1, 2, \ldots, m \right\}.$$

Therefore, by [24, Corollary 23.5.3],

$$\partial \vartheta_\kappa(y) = \operatorname{argmax}\left\{ \sum_{i=1}^{m} \lambda_i y_i : \sum_{i=1}^{m} \lambda_i = \kappa,\ 0 \leqslant \lambda_i \leqslant 1, i = 1, 2, \ldots, m \right\}.$$

It is easily shown that the set on the right side of the last equality is exactly

$$\left\{ \sum_{j \in J(y)} \rho_j e_j : \sum_{j \in J(y)} \rho_j = \kappa,\ 0 \leqslant \rho_j \leqslant 1 \text{ for } j \in J(y), \rho_j = 0 \text{ for } j \notin J(y) \right\},$$

where $\{e_1, e_2, \ldots, e_m\}$ denote the canonical basis of $\mathbb{R}^m$ and

$$J(y) = \{ j \in \{1, 2, \ldots, m\} | y_j \geqslant y_{[\kappa]} \}.$$

Thus, from [16, Theorem 4.3.1], we immediately obtain (16). □

Lemma 3. *Let $\{(w^k, x^k)\}$ be the sequence generated by Algorithm 1. Then, any limit points of $\{x^k\}$ are optimal solutions to problem (1).*

Proof. Let (w^*, x^*) be a limit point of the sequence $\{(w^k, x^k)\}$. Without loss of generality, we assume that $\{(w^k, x^k)\} \to (w^*, x^*)$ when k tends to $+\infty$. We now prove that x^* is an optimal solution to problem (1). First, from (10) and the fact that (w^k, x^k) is a solution of (14), it follows that

$$\Phi_\kappa(x^k) \leqslant \Phi(w^k, x^k, \varepsilon_k) \leqslant \Phi_\kappa(x^k) + m(\ln 2 + 1)\varepsilon_k. \tag{17}$$

By the continuity of $f_i(x)$, we thus have

$$\Phi_\kappa(x^*) = \lim_{k \to +\infty} \Phi(w^k, x^k, \varepsilon_k) = \kappa w^* + \sum_{i=1}^{m} \max\{0, f_i(x^*) - w^*\},$$

which can be rewritten as

$$\Phi_\kappa(x^*) = \kappa w^* + \sum_{i \in I(x^*)} \max\{0, f_i(x^*) - w^*\} + \sum_{i \notin I(x^*)} \max\{0, f_i(x^*) - w^*\}.$$

Here, $I(x^*)$ is defined by (15). The last equation implies that

$$f_i(x^*) - w^* \leqslant 0 \text{ for all } i \notin I(x^*). \tag{18}$$

In addition, from the fact that (w^k, x^k) is a solution of (14), we have

$$\nabla \Phi(w^k, x^k, \varepsilon_k) = \begin{pmatrix} \kappa - \sum_{i=1}^{m} \lambda_i^k \\ \sum_{i=1}^{m} \frac{\lambda_i^k \omega_i (x^k - a_i)}{\sqrt{\|x - a_i\|^2 + \varepsilon^2}} \end{pmatrix} = 0, \tag{19}$$

where $\lambda_i^k = \lambda_i(w^k, x^k, \varepsilon_k)$ for $i = 1, 2, \ldots, m$. Combining with the definition of $\lambda_i(w^k, x^k, \varepsilon_k)$ in (12), it is clear that $\sum_{i=1}^{m} \lambda_i^k = \kappa$ and $0 < \lambda_i^k < 1$ for $i = 1, 2, \ldots, m$. This means that the sequence $\{\lambda_i^k\}$ for every $i = 1, 2, \ldots, m$ has a convergent subsequence. Without loss of generality, we suppose that

$$\lim_{k \to +\infty} \lambda_i^k = \lambda_i^*, \quad i = 1, 2, \ldots, m.$$

Then,

$$\sum_{i=1}^{m} \lambda_i^* = \kappa, \quad 0 \leqslant \lambda_i^* \leqslant 1 \quad \text{for } i = 1, 2, \ldots, m. \tag{20}$$

Next we prove $\lambda_i^* = 0$ for $i \notin I(x^*)$. By (18), we consider the following two cases to prove it.

Case 1: there exits an index $i_0 \notin I(x^*)$ such that $f_{i_0}(x^*) - w^* = 0$. In this case, there must hold

$$f_{[1]}(x^*) - w^* > 0, f_{[2]}(x^*) - w^* > 0, \ldots, f_{[\kappa]}(x^*) - w^* > 0.$$

By the continuity of $f_i(x)$, we have for all sufficiently large k

$$f_{[1]}(x^k, \varepsilon_k) - w^k > 0, f_{[2]}(x^k, \varepsilon_k) - w^k > 0, \ldots, f_{[\kappa]}(x^k, \varepsilon_k) - w^k > 0.$$

This implies that

$$\lim_{k \to +\infty} \frac{\exp[(f_{[l]}(x^k, \varepsilon_k) - w^k)/\varepsilon_k]}{1 + \exp[(f_{[l]}(x^k, \varepsilon_k) - w^k)/\varepsilon_k]} = 1, \quad l = 1, 2, \ldots, \kappa.$$

Thus, from (17) and the definition of λ_i^k, we have

$$\sum_{i \notin I(x^*)} \lambda_i^* = \kappa - \lim_{k \to +\infty} \sum_{i \in I(x^*)} \frac{\exp[(f_i(x^k, \varepsilon_k) - w^k)/\varepsilon_k]}{1 + \exp[(f_i(x^k, \varepsilon_k) - w^k)/\varepsilon_k]} = 0.$$

Consequently, $\lambda_i^* = 0$ for $i \notin I(x^*)$.

Case 2: $f_i(x^*) - w^* < 0$ for all $i \notin I(x^*)$. Now, when k is sufficiently large,

$$f_i(x^k, \varepsilon_k) - w^k < 0 \quad \text{for all } i \notin I(x^*)$$

due to the continuity of $f_i(x)$. Thus, from the definition of λ_i^k, we readily have $\lambda_i^* = 0$ for $i \notin I(x^*)$.

Thus, combining with the Eq. (20), we have

$$\sum_{i \in I(x^*)} \lambda_i^* = \kappa, \quad 0 \leqslant \lambda_i^* \leqslant 1 \text{ for } i \in I(x^*), \text{ and } \lambda_i^* = 0 \text{ for } i \notin I(x^*). \tag{21}$$

Note that for $i \in I(x^*)$,

$$v_i^* = \lim_{k \to +\infty} \frac{\omega_i(x^k - a_i)}{\sqrt{\|x^k - a_i\|^2 + \varepsilon_k^2}} \in \partial f_i(x^*).$$

Therefore, it follows from the Eqs. (19) and (21) that

$$\sum_{i \in I(x^*)} \lambda_i^* v_i^* = \lim_{k \to +\infty} \sum_{i=1}^{m} \frac{\lambda_i^k \omega_i(x^k - a_i)}{\sqrt{\|x - a_i\|^2 + \varepsilon^2}} = 0.$$

Compared with (16), this indicates that $0 \in \partial \Phi_\kappa(x^*)$, and accordingly x^* is an optimal solution to problem (1). Here, we complete the proof of lemma. □

Theorem 1. *Let* $\{x^k\}$ *be the sequence generated by Algorithm* 1. *If* x^* *is the unique optimal solution of problem* (1), *then we have* $\lim_{k \to +\infty} x^k = x^*$.

Proof. For any $k \geqslant 1$, by Lemma 1,

$$\Phi(w^1, x^1, \varepsilon_1) > \Phi(w^1, x^1, \varepsilon_k) \geqslant \Phi(w^k, x^k, \varepsilon_k) \geqslant \Phi_\kappa(x^k). \tag{22}$$

Consider that $\Phi_\kappa(x)$ is coercive, and so the level set $L = \{x \in \mathbb{R}^n | \Phi_\kappa(x) \leqslant \Phi(w^1, x^1, \varepsilon_1)\}$ is bounded. However, from (22), we have $\{x^k\} \subseteq L$. This shows that the sequence $\{x^k\}$ is bounded. Since x^* is the unique solution of problem (1), we have from Lemma 2 that $\lim_{k \to +\infty} x^k = x^*$. □

In practice, we would stop Algorithm 1 once some stopping criteria are satisfied. Moreover, we will use a first-order, or gradient-based, unconstrained minimization algorithm to solve the problem (14) in Step 1. In what follows, we describe a more practical version of Algorithm 1 by choosing a limited-memory BFGS algorithm to solve the problem (14) in Step 1 because this algorithm can solve very large unconstrained optimization problem efficiently.

Algorithm 2. Let $\sigma \in (0,1)$, $\tau_1, \tau_2 > 0$, $(\hat{w}^0, \hat{x}^0) \in \mathbb{R}^{n+1}$ and $\varepsilon_0 > 0$ be given. Set $k := 0$.

For $k = 0, 1, 2, \ldots,$ until $\varepsilon_k \leqslant \tau_1$, **do**

1. Use a version of the limited-memory BFGS algorithm with (w^k, x^k) as the starting point to solve (14) approximately, and obtain an (w^k, x^k) such that $\|\nabla \Phi(w^k, x^k, \varepsilon_k)\| \leqslant \tau_2$.
2. Set $\varepsilon_{k+1} = \sigma\varepsilon_k$, $(\hat{w}^k, \hat{x}^k) = (w^{k+1}, x^{k+1})$, and then go back to Step 1.

End

4. Second-order cone program reformulation

In this section, we will reformulate the single-facility κ-centrum problem (1) as a standard primal second-order cone program. First, from the discussions in the second paragraph of Section 2, we know that problem (1) is equivalent to the optimization problem

$$\begin{aligned} \min_{w,v,\eta} \quad & \kappa w + \sum_{i=1}^{m} \eta_i \\ \text{s.t.} \quad & \|\omega_i(v - a_i)\| \leqslant \eta_i + w, \quad i = 1, 2, \ldots, m, \\ & \eta_i \geqslant 0 \end{aligned} \tag{23}$$

where $v = (v_1, \ldots, v_n)^{\mathrm{T}} \in \mathbb{R}^n$. This problem can be rewritten as

$$\begin{aligned} \min \ & (\kappa - m)w + \sum_{i=1}^{m} \omega_i t_i \\ \text{s.t.} \ & \sqrt{(v_1 - a_{11})^2 + (v_2 - a_{12})^2 + \cdots + (v_n - a_{1n})^2} \leqslant t_1, \\ & \sqrt{(v_1 - a_{21})^2 + (v_2 - a_{22})^2 + \cdots + (v_n - a_{2n})^2} \leqslant t_2, \\ & \qquad \vdots \\ & \sqrt{(v_1 - a_{m1})^2 + (v_2 - a_{m2})^2 + \cdots + (v_n - a_{mn})^2} \leqslant t_m, \\ & \omega_i t_i - w \geqslant 0, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{24}$$

Let

$$\begin{aligned} & \omega_i t_i - w = \upsilon_i, \quad i = 1, 2, \ldots, m; \\ & v_j - a_{ij} = u_{ij}, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n. \end{aligned} \tag{25}$$

Then the constraints of problem (24) become

$$\begin{cases} \omega_1 t_1 - \upsilon_1 = \omega_2 t_2 - \upsilon_2 = \cdots = \omega_m t_m - \upsilon_m = w, \\ \upsilon_i \geqslant 0, \quad i = 1, 2, \ldots, m, \end{cases}$$

$$\begin{cases} u_{11}^2 + u_{12}^2 + \cdots + u_{1n}^2 \leqslant t_1^2, \quad t_1 \geqslant 0, \\ u_{21}^2 + u_{22}^2 + \cdots + u_{2n}^2 \leqslant t_2^2, \quad t_2 \geqslant 0, \\ \qquad\qquad \vdots \qquad\qquad \vdots \qquad \vdots \\ u_{m1}^2 + u_{m2}^2 + \cdots + u_{mn}^2 \leqslant t_m^2, \quad t_m \geqslant 0, \end{cases}$$

and

$$\begin{cases} u_{11}+a_{11}=u_{21}+a_{21}=\cdots=u_{m1}+a_{m1}, \\ u_{12}+a_{12}=u_{22}+a_{22}=\cdots=u_{m2}+a_{m2}, \\ \quad\vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\ u_{1n}+a_{1n}=u_{2n}+a_{2n}=\cdots=u_{mn}+a_{mn}. \end{cases}$$

Let

$$z_i=(t_i,u_{i1},u_{i2},\ldots,u_{in})\in\mathbb{R}^{n+1},\quad c_i=\left(\frac{\kappa}{m}\omega_i;0_n;1-\frac{\kappa}{m}\right)\in\mathbb{R}^{n+2},\quad i=1,2,\ldots,m \tag{26}$$

and define the second-order cone

$$\mathscr{K}^{n+1}=\{(\xi_1;\xi_2)|\xi_1\in\mathbb{R}_+,\xi_2\in\mathbb{R}^n,\quad\text{and}\quad\|\xi_2\|\leqslant\xi_1\}.$$

Then,

$$x_i:=(z_i;\upsilon_i)\in\mathscr{K}^{n+1}\times\mathbb{R}_+,\quad i=1,2,\ldots,m,$$

and furthermore, it follows from (25) that

$$c_1^{\mathrm{T}}x_1+c_2^{\mathrm{T}}x_2+\cdots+c_m^{\mathrm{T}}x_m=(k-m)w+\sum_{i=1}^{m}\omega_i t_i.$$

Therefore, the problem in (24) turns into the form of

$$\min\quad\sum_{i=1}^{m}c_i^{\mathrm{T}}x_i$$

$$\text{s.t.}\quad\begin{cases} (\omega_1t_1-\omega_2t_2)-(\upsilon_1-\upsilon_2)=0, \\ (\omega_1t_1-\omega_3t_3)-(\upsilon_1-\upsilon_3)=0, \\ \qquad\vdots \\ (\omega_1t_1-\omega_mt_m)-(\upsilon_1-\upsilon_m)=0, \\ u_{11}-u_{21}=a_{21}-a_{11}, \\ u_{11}-u_{31}=a_{31}-a_{11}, \\ \qquad\vdots \\ u_{11}-u_{m1}=a_{m1}-a_{11}, \\ u_{12}-u_{22}=a_{22}-a_{12}, \\ u_{12}-u_{32}=a_{32}-a_{12}, \\ \qquad\vdots \\ u_{12}-u_{m2}=a_{m2}-a_{12}, \\ \qquad\vdots \\ u_{1n}-u_{2n}=a_{2n}-a_{1n}, \\ u_{1n}-u_{3n}=a_{3n}-a_{1n}, \\ \qquad\vdots \\ u_{1n}-u_{mn}=a_{mn}-a_{1n}, \\ (x_1;x_2;\ldots;x_m)\in(\mathscr{K}^{n+1}\times\mathbb{R}_+)\times\cdots\times(\mathscr{K}^{n+1}\times\mathbb{R}_+). \end{cases} \tag{27}$$

Let

$$\begin{aligned} &N = m(n+2), \quad N_1 = (m-1)(n+1), \quad \mathscr{K} = (\mathscr{K}^{n+1} \times \mathbb{R}_+) \times \cdots \times (\mathscr{K}^{n+1} \times \mathbb{R}_+), \\ &x = (x_1; \ldots; x_m) \in \mathbb{R}^N, \quad c = (c_1; \ldots; c_m) \in \mathbb{R}^N, \quad b = (0_{m-1}; b_1; \ldots; b_n) \in \mathbb{R}^{N_1}, \end{aligned} \tag{28}$$

where

$$b_j = (a_{2j} - a_{1j}, a_{3j} - a_{1j}, \ldots, a_{mj} - a_{1j})^{\mathrm{T}} \in \mathbb{R}^{m-1}, \quad \text{for} \quad j = 1, 2, \ldots, n.$$

Then problem (27) can be recast into a standard primal SOCP:

$$\begin{aligned} &\min \quad c^{\mathrm{T}} x \\ &\text{s.t.} \quad Ax = b, \quad x \in \mathscr{K}, \end{aligned} \tag{29}$$

where

$$A = \left[\begin{array}{cccc} B_{11} & B_{12} & \cdots & B_{1m} \\ B_{21} & B_{22} & \cdots & B_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)1} & B_{(m-1)2} & \cdots & B_{(m-1)m} \\ \hdashline A_{11} & A_{12} & \cdots & A_{1m} \\ A_{21} & A_{22} & \cdots & A_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ A_{(m-1)1} & A_{(m-1)2} & \cdots & A_{(m-1)m} \\ \hdashline \vdots & \vdots & \cdots & \vdots \\ \hdashline A_{(n(m-1)-m+2)1} & A_{(n(m-1)-m+2)2} & \cdots & A_{(n(m-1)-m+2)m} \\ A_{(n(m-1)-m+1)1} & A_{(n(m-1)-m+1)2} & \cdots & A_{(n(m-1)-m+1)m} \\ \vdots & \vdots & \cdots & \vdots \\ A_{(n(m-1))1} & A_{(n(m-1))2} & \cdots & A_{(n(m-1))m} \end{array}\right] \in \mathbb{R}^{N_1 \times N} \tag{30}$$

in which the element B_{kl} is a row vector in $\mathbb{R}^{n+2}$ defined by

$$B_{kl} = \begin{cases} (\omega_1, 0, \ldots, 0, -1), & \text{if } l = 1, \\ (-\omega_l, 0, \ldots, 0, 1), & \text{if } l = k+1, \\ 0_{n+2}, & \text{otherwise}, \end{cases}$$

and the element A_{kl} is a row vector in $\mathbb{R}^{n+2}$ with the $(\lceil k/m \rceil + 2)$th element being 1 and the others 0 if $l = 1$, otherwise if $l = k+1$ the $(\lceil k/m \rceil + 2)$th element being -1 and the others 0, and otherwise it is a zero vector in $\mathbb{R}^{n+2}$. Here, $\lceil k/m \rceil$ is the largest positive integer not over than k/m.

It is not difficult to verify that A defined in (30) has a full row rank N_1. We here want to point out that the similar reformulation techniques developed as above were also used by [7,17] for facility location problems in $\mathbb{R}^2$, where the resulting SOCP problems are solved by a merit function approach and an interior point method, respectively. In the next section, we will develop an algorithm for problem (1) by following the same line as [7].

5. Algorithm based on the SOCP reformulation

In this section, we use the Fischer–Burmeister merit function developed by [8] for SOCCPs to transform the KKT system of the SOCP (29) into an equivalent unconstrained smooth minimization problem, and then based on this minimization problem, present an algorithm for solving problem (1).

The KKT optimality conditions of (29), which are sufficient but not necessary for optimality, are

$$\begin{aligned} &\langle x, y\rangle = 0, \quad x \in \mathcal{K}, y \in \mathcal{K}, \\ &Ax = b, \quad y = c - A^{\mathrm{T}}\zeta_d \quad \text{forsome } \zeta_d \in \mathbb{R}^{N_1}, \end{aligned} \tag{31}$$

where $y = (y_1; \ldots; y_m)$ and $y_i = (s_i; w_i)$ with $s_i \in \mathbb{R}^{n+1}$ and $w_i \in \mathbb{R}_+$ for $i = 1, 2, \ldots, m$. Choose any $d \in \mathbb{R}^N$ satisfying $Ad = b$. (If no such d exists, then (29) has no feasible solution). Let $A_{\mathcal{N}} \in \mathbb{R}^{N\times(N-N_1)}$ be any matrix whose columns span the null space of A. Then x satisfies $Ax = b$ if and only if $x = d + A_{\mathcal{N}}\zeta_p$ for some $\zeta_p \in \mathbb{R}^{N-N_1}$. Thus, the KKT optimality conditions in (31) can be rewritten as the following SOCCP:

$$\begin{aligned} &\langle x, y\rangle = 0, \quad x \in \mathcal{K}, \quad y \in \mathcal{K}, \\ &x = F(\zeta), \quad y = G(\zeta), \end{aligned} \tag{32}$$

where

$$\zeta = (\zeta_p; \zeta_d), \quad F(\zeta) := d + A_{\mathcal{N}}\zeta_p, \quad G(\zeta) := c - A^{\mathrm{T}}\zeta_d. \tag{33}$$

Alternatively, consider that any $\zeta \in \mathbb{R}^N$ can be decomposed into the sum of its orthogonal projection on the column space of A^{T} and the null space of A, so the following form can be used in place of (33)

$$F(\zeta) := d + (I - A^{\mathrm{T}}(AA^{\mathrm{T}})^{-1}A)\zeta, \quad G(\zeta) := c - A^{\mathrm{T}}(AA^{\mathrm{T}})^{-1}A\zeta. \tag{34}$$

In the sequel, unless otherwise stated, $F(\zeta)$ and $G(\zeta)$ are both defined by (34).

In [8], Chen extended the merit function, the squared norm of the Fischer–Burmeister function for the nonlinear complementarity problems, to SOCCPs, and then developed a unconstrained optimization technique for SOCCPs by use of the merit function. For any $u = (u_1, u_2), v = (v_1, v_2) \in \mathbb{R} \times \mathbb{R}^n$, define their *Jordan product* associated with $\mathcal{K}^{n+1}$ by

$$u \circ v := (\langle u, v\rangle, v_1 u_2 + u_1 v_2).$$

The identity element under this product is $(1, 0, \ldots, 0) \in \mathbb{R}^{n+1}$. Write u^2 to mean $u \circ u$ and $u + v$ to mean the usual componentwise addition of vectors. It is well known that $u^2 \in \mathcal{K}^{n+1}$ for all $u \in \mathbb{R}^{n+1}$. Moreover, if $u \in \mathcal{K}^{n+1}$, there is a unique vector in $\mathcal{K}^{n+1}$, denoted by $u^{1/2}$, such that $(u^{1/2})^2 = u^{1/2} \circ u^{1/2}$. Then,

$$\phi(u, v) := (u^2 + v^2)^{1/2} - u - v \tag{35}$$

is well-defined for all $(u, v) \in \mathbb{R}^{n+1} \times \mathbb{R}^{n+1}$ and maps $\mathbb{R}^{n+1} \times \mathbb{R}^{n+1}$ to $\mathbb{R}^{n+1}$. It was shown in [15] that

$$\phi(u, v) = 0 \Longleftrightarrow \langle u, v\rangle = 0, \quad u \in \mathcal{K}^{n+1}, \quad v \in \mathcal{K}^{n+1}.$$

Note that $\mathcal{K}^1$ coincides with the set of nonnegative reals $\mathbb{R}_+$, and in that case $\phi(u, v)$ in (35) reduces to the Fischer–Burmeister NCP-function [12,13]. Thus,

$$\psi(x, y) := \frac{1}{2}\sum_{i=1}^{m}\left(\|\phi(z_i, s_i)\|^2 + \phi^2(v_i, w_i)\right) \tag{36}$$

is a merit function for the SOCCP (32), where $z_i, s_i \in \mathbb{R}^{n+1}$ and $v_i, w_i \in \mathbb{R}_+$. Consequently, (32) can rewritten as an equivalent global minimization problem as below:

$$\min_{\zeta\in\mathbb{R}^N} \Psi(\zeta) = \psi(F(\zeta), G(\zeta)). \tag{37}$$

Moreover, we know from [8] that ψ is smooth and every stationary point of (32) solves the SOCP (29).

Now we establish an algorithm for the problem (1) by solving the unconstrained smooth minimization problem (32) with a limited-memory BFGS method.

Algorithm 3. Let $\tau_1, \tau_2, \tau_3 > 0$ and $\zeta^0 \in \mathbb{R}^N$ be given. Generate a steepest descent direction $\Delta^0 = -\nabla\Psi(\zeta^0)$ and seek a suitable stepsize α_0. Let

$$\zeta^1 := \zeta^0 + \alpha_0\Delta^0, \quad x^1 := F(\zeta^1), \quad y^1 := G(\zeta^1).$$

For $k = 1, 2, \ldots,$ until $\Psi(\zeta^k) \leqslant \tau_1$ and $|(x^k)^{\mathrm{T}} y^k| \leqslant \tau_2$, **do**
  If $(\nabla\Psi(\zeta^k) - \nabla\Psi(\zeta^{k-1}))^{\mathrm{T}}(\zeta^k - \zeta^{k-1}) \leqslant \tau_3\|\zeta^k - \zeta^{k-1}\| \cdot \|\nabla\Psi(\zeta^k) - \nabla\Psi(\zeta^{k-1})\|$, **then**
    $\Delta^k = -\nabla\Psi(\zeta^k)$;
  else
    Δ^k is generated by the limited-memory BFGS method.
  End
Set $\zeta^k := \zeta^k + \alpha_k\Delta^k, x^k := F(\zeta^k), y^k := G(\zeta^k)$, where α_k is a stepsize. Let $k := k + 1$.
End

Remark 1. Suppose that ζ^* is the final iteration generated by Algorithm 3 and $x^* = F(\zeta^*)$. Then, $x^*(2 : n + 1) + a_1$ is the optimal solution of problem (1) and the optimal value is $c^{\mathrm{T}}x^*$.

6. Numerical experiments

We implemented Algorithm 2 in Section 3 and Algorithm 3 in Section 5 with our codes and compared them with SeDuMi 1.05 [27] (A high quality software packages with Matlab interface for solving SOCP and semidefinite programming problems). Our computer codes are all written in Matlab, including the evaluation of $\psi(x, y)$ and $\nabla\psi(x, y)$ in Algorithm 3. All the numerical experiments were done at a PC with CPU of 2.8 GHz and RAM of 512 MB. To solve the unconstrained minimization problem (14) in Algorithm 2 and generate the direction Δ^k and the suitable stepsize in Algorithm 3, we choose a limited-memory BFGS algorithm with Armijo line-search and 5 limited-memory vector-updates [5], where for the scaling matrix $H^0 = \gamma I_N$ we use the recommended choice of $\gamma = p^{\mathrm{T}}q/q^{\mathrm{T}}q$ [18, P. 226], where $p := \zeta - \zeta^{\mathrm{old}}$ and $q := \nabla\Psi(\zeta) - \nabla\Psi(\zeta^{\mathrm{old}})$. This choice is found to work better than the choice used by [8] for our problems. To evaluate F and G in Algorithm 3, we use LU factorization of AA^{T}. In particular, given such a factorization $LU = AA^{\mathrm{T}}$, we can compute $x = F(\zeta)$ and $s = G(\zeta)$ for each ζ via two matrix-vector multiplications and two forward/backward solves:

$$Lu = A\zeta, \quad Uv = \xi, \quad w = A^{\mathrm{T}}v, \quad x = d + \zeta - w, \quad s = c - w. \tag{38}$$

For the vector d satisfying $Ad = b$, we compute it as a solution of $\min_d\|Ad - b\|$ using Matlab's linear least square solver "lsqlin".

Throughout the computational experiments, we use the following parameters in Algorithm 2:

$$\varepsilon_0 = 1, \quad \sigma = 0.1, \quad \tau_1 = 1.0e-6, \quad \tau_2 = 1.0e-3.$$

For Algorithm 3, we use the following parameters:

$$\tau_1 = 1.0e-8, \quad \tau_2 = 1.0e-5, \quad \tau_3 = 1.0e-4.$$

For SeDuMi, we use all the default values except that the parameter **eps** is set to be 1.0e–6. The starting points for Algorithms 2 and 3 are chosen to be $\hat{x}^0 = 0_n, \hat{w}^0 = 0$ and $\zeta^0 = 0_N$, respectively.

The test problems are generated randomly by the following pseudo-random sequence:

$$\psi_0 = 7, \quad \psi_{i+1} = (445\psi_i + 1) \bmod 4096, \quad i = 1, 2, \ldots,$$
$$\bar{\psi}_i = \psi_i/40.96, \quad i = 1, 2, \ldots,$$

The elements of a_i for $i = 1, 2, \ldots, m$ are successively set to $\bar{\psi}_1, \bar{\psi}_2, \ldots,$ in the order:

$$a_1(1), a_1(2), \ldots, a_1(n), a_2(1), a_2(2), \ldots, a_2(n), \ldots, a_m(1), a_m(2), \ldots, a_m(n),$$

and the weight ω_i is set to be $a_i(1)/10$ if $\mathrm{mod}(i, 10) = 0$, and otherwise ω_i is set to be 1.

The numerical results are summarized in Tables 1–3. In these tables, n and m specify the problem dimensions, **Obj.** denotes the objective value of (1) at the final iteration, **Iter** indicates the iteration number and **Time** represents the CPU time in seconds for solving each problem, and for Algorithm 3, it also includes the time to make the LU decomposition of AA^{T} and find the feasible point d.

Table 1
Numerical results for the problems with different κ $(m = 50,\ n = 1000)$

κ	Algorithm 2			Algorithm 3			SeDuMi		
	Obj.	Iter	Time	Obj.	Iter	Time	Obj.	Iter	Time
1	5.941976e+3	195	2.65	5.941976e+3	2118	1434	5.941978e+4	12	48.3
5	2.509660e+4	24	0.28	2.509660e+4	344	229.5	2.509660e+4	10	50.1
10	3.021423e+4	258	3.39	3.021423e+4	667	445.7	3.021423e+4	13	51.6
15	3.516542e+4	364	4.40	3.516542e+4	570	396.5	3.516542e+4	13	50.7
25	4.480229e+4	392	4.79	4.480228e+4	458	317.1	4.480229e+4	13	54.3
35	5.418417e+4	425	4.90	5.418417e+4	361	253.8	5.418417e+4	13	50.6
40	5.880483e+4	414	4.90	5.880483e+4	381	270.0	5.880484e+4	13	51.5
45	6.338237e+4	240	3.46	6.338237e+4	438	326.1	6.338238e+4	13	51.4
50	6.790952e+4	31	0.33	6.790952e+4	354	274.2	6.790952e+4	7	27.7

Table 2
Performance comparison of Algorithms 2, 3 and SeDuMi

Problem			Algorithm 2			Algorithm 3			SeDuMi		
m	n	κ	Obj.	Iter	Time	Obj.	Iter	Time	Obj.	Iter	Time
100	100	10	1.441721e+4	109	1.15	1.441720e+4	993	235.3	1.441721e+4	15	55.7
100	200	10	1.732435e+4	92	1.14	1.732435e+4	1089	453.1	1.732435e+4	16	208.9
100	300	10	2.156258e+4	58	0.81	2.156258e+4	2216	1454	2.156258e+4	14	662.3
100	400	10	1.841160e+4	177	2.51	1.841159e+4	1851	1618	1.841160e+4	14	2555
100	500	10	3.492155e+4	40	0.67	3.492156e+4	704	710.5	*	*	*
100	600	10	3.358388e+4	154	3.51	3.358388e+4	1930	2370	*	*	*
100	700	10	3.435271e+4	45	0.78	3.435271e+4	2135	3064	*	*	*
100	800	10	3.512760e+4	57	1.20	3.512760e+4	2710	4404	*	*	*
100	900	10	3.958261e+4	26	0.53	3.958261e+4	2152	4056	*	*	*
100	1000	10	5.960709e+4	27	0.59	5.960709e+4	258	570.9	*	*	*

Table 3
Performance of Algorithm 2 on very large problems

m	n	κ	Obj.	iter	CPU	m	n	κ	Obj.	iter	CPU
1000	1000	10	83.2309391e+4	493	173.8	1000	2000	10	1.17181049e+5	568	515.1
1000	4000	10	1.59968897e+5	424	1126	1000	6000	10	2.02300923e+5	512	2158
1000	8000	10	2.17901326e+5	295	1498	1000	10000	10	2.61377885e+5	518	3206
2000	1000	20	1.71789468e+5	368	286.2	2000	2000	20	2.36133844e+5	319	633.2
2000	3000	20	2.94584768e+5	356	1327	2000	4000	20	3.20657339e+5	515	3430
2000	5000	20	3.77768347e+5	479	3508	2000	6000	20	4.08304039e+5	664	5058
3000	1000	30	2.56936690e+5	384	486.7	3000	2000	30	3.55199292e+5	630	2020
3000	3000	30	4.41767230e+5	462	2422	3000	4000	30	4.80986010e+5	404	4214
4000	1000	40	3.42205909e+5	617	1039	4000	3000	40	5.90447512e+5	405	3957

The results in Table 1 show how the iteration number of Algorithms 2 and 3 and SeDuMi varies with κ for the problems of the same dimension ($m = 50$ and $n = 1000$). We can see from this table that the value of κ has a remarkable influence on their iteration number, and when κ closes to m, they will decrease, but when κ closes to 1, they will increase, and especially that of Algorithms 2 increase greatly.

The results listed in Tables 1 and 2 show that the algorithms presented in this paper perform very well and they are able to obtain good accuracy for all test problems. Particularly, Algorithm 2 consistently uses less CPU time than Algorithm 3 and SeDuMi. For the moderate problems where n and m are in the order of hundreds, Algorithm 3 is comparable with SeDuMi, and moreover, we can see from Table 2 that the CPU time of Algorithm 3 increases slower than that of SeDuMi with the dimension of problem increasing. In addition, we

should point out that the evaluations of $\psi(x, y)$ and $\nabla\psi(x, y)$ coded in Matlab increased the CPU time of Algorithm 3 greatly since it is a main part of the algorithm.

The results in Table 3 show that Algorithm 2 can solve very large problems in a reasonable amount of CPU time. However, Algorithm 3 and SeDuMi failed for these problems due to excessive CPU time or memory requirement.

We conclude that Algorithm 2 is better than Algorithm 3 and SeDuMi for very large problems since it consumes less memory. For moderate problems, Algorithm 3 and SeDuMi are comparable.

7. Conclusions

In this paper, we have presented two kinds of unconstrained optimization techniques for the single-facility Euclidean κ-centrum location problem based on a simplified unconstrained formulation and a SOCP reformulation, respectively. The first method is actually a primal one whereas the second is a primal-dual one. Preliminary numerical experiments show that the two methods can obtain desirable accuracy for all test problems and the first method is extremely efficient for those problems where n is large and m is moderate. Though the two methods are developed for the single-facility location problem, they can be extended to the multi-facilities Euclidean κ-centrum location problem.

Acknowledgement

The author's work is partially supported by the Doctoral Starting-up Foundation (B13B6050640) of GuangDong Province. The author's work is partially supported by National Science Council of Taiwan.

References

[1] K.D. Andersen, E. Christiansen, A.R. Conn, M.L. Overton, An efficient primal-dual interior-point method for minimizing a sum of Eucidean norms, SIAM Journal on Optimization 1 (2000) 243–262.
[2] F. Alizadeh, D. Goldfarb, Second-order cone programming, Mathematical Programming, Series B 95 (2003) 3–51.
[3] G. Andreatta, F.M. Mason, κ-Eccentricity and absolute κ-centrum of tree, European Journal of Operational Research 19 (1985) 114–117.
[4] G. Andreatta, F.M. Mason, Properties of κ-centrum in a network, Networks 15 (1985) 21–25.
[5] R.H. Byrd, P. Lu, J. Nocedal, C. Zhu, A limited memory algorithm for bound constrained optimization, SIAM Journal of Scientific Computing 16 (1995) 1190–1208.
[6] C.H. Chen, O.L. Mangasarian, A class of smoothing functions for nonlinear and mixed complementarity problems, Computational Optimization and Applications 5 (1996) 97–138.
[7] J.-S. Chen, A merit function approach for a facility location problem, 2005.
[8] J.-S. Chen, P. Tseng, An unconstrained smooth minimization reformulation of the second-order cone complementarity problem, Mathematical Programming, Series B 95 (2005) 3–51.
[9] Z. Drezner, Facility Location. A Survey of Applications and Methods, Springer-Verlag, Berlin, 1995.
[10] Z. Drezner, H. Hamacher, Facility Location: Applications and Theory, Springer-Verlag, Berlin, 2002.
[11] J.W. Eyster, J.A. White, W.W. Wierwille, On solving multifacility location problems using a hyperboloid approximation procedure, AIIE Transaction 5 (1973) 1–6.
[12] A. Fischer, A special Newton-type optimization methods, Optimization 24 (1992) 269–284.
[13] A. Fischer, Solution of the monotone complementarity problem with locally Lipschitzian functions, Mathematical Programming 76 (1997) 513–532.
[14] R.L. Francis, L.F. McGinnis Jr., J.A. White, Facility Layout and Location: An Analytic Approach, Prentice-Hall, Englewood Cliffs, NJ, 1991.
[15] M. Fukushima, Z.Q. Luo, P. Tseng, Smoothing functions for second-order cone complementarity problems, SIAM Journal on Optimization 12 (2002) 436–460.
[16] J.B. Hiriart-Urruty, C. Lemarèchal, Convex Analysis and Minimization Algorithm, Springer-Verlag, Berlin Heidelberg, 1993.
[17] Y.-J. Kuo, H.D. Mittelmann, Interior point methods for second-order cone programming and OR applications, Computational Optimization and Applications 28 (2004) 255–285.
[18] J. Nocedal, S.J. Wright, Numerical Optimization, Springer-Verlag, New York, 1999.
[19] L.Q. Qi, G.L. Zhou, A smoothing Newton method for minimizing a sum of Euclidean norms, SIAM Journal on Optimization 11 (2000) 389–410.
[20] L.Q. Qi, D.F. Sun, G.L. Zhou, A primal-dual algorithm for minimizing a sum of Euclidean norms, Journal of Computational and Applied Mathematics 138 (2002) 34–63.

[21] M.S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, Applications of second-order cone programming, Linear Algebra and its Applications 284 (1998) 193–228.
[22] W. Ogryczak, A. Tamir, Minimizing the sum of the k largest functions in linear time, Information Processing Letters 85 (2003) 117–122.
[23] W. Ogryczak, M. Zawadzki, Conditional median: a parametric solution concept for location problems, Annals of Operations Research 110 (2002) 167–181.
[24] R.T. Rockafella, Convex Analysis, Princeton University Press, Princeton, New Jersey, 1970.
[25] P.J. Slater, Centers to centroids in a graph, Journal of Graph Theory 2 (1978) 209–222.
[26] S.Y. Shi, Smooth convex approximations and its applications. Master thesis, Department of Mathematics, National University of Singapore, 2004.
[27] J.F. Sturm, Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, Optimization methods & software 11 (12) (1999) 625–653.
[28] A. Tamir, The k-centrum multi-facility location problem, Discrete Applied Mathematics 109 (2001) 293–307.
[29] G.L. Xue, J.B. Rosen, P.M. Pardalos, An polynomial time dual algorithm for the Euclidean multifacility location problem, Operations Research Letters 18 (1996) 201–204.
[30] G.L. Xue, Y. Ye, An efficient algorithm for minimizing a sum of Euclidean norms with applications, SIAM Journal on Optimization 7 (1997) 1017–1036.