Minimizing Differences of Convex Functions with Applications to Facility Location and Clustering

Nguyen Mau Nam · R. Blake Rector · Daniel Giles

Abstract In this paper, we develop algorithms to solve generalized Fermat–Torricelli problems with both positive and negative weights and multifacility location problems involving distances generated by Minkowski gauges. We also introduce a new model of clustering based on squared distances to convex sets. Using the Nesterov smoothing technique and an algorithm for minimizing differences of convex functions introduced by Tao and An, we develop effective algorithms for solving these problems. We demonstrate the algorithms with a variety of numerical examples.

Keywords Difference of convex functions · Nesterov smoothing technique · Fermat–Torricelli problem · Multifacility location · Clustering

Mathematics Subject Classification 49J52 · 49J53 · 90C31

1 Introduction

To solve the classical Fermat–Torricelli problem, one finds a point that minimizes the sum of the Euclidean distances to three points in the plane. This problem was
introduced by Pierre de Fermat in the seventeenth century and originally solved by Evangelista Torricelli. A more general model asks for a point that minimizes the sum of the Euclidean distances to a finite number of points in a finite dimensional Euclidean space. In spite of its simplicity, this problem has been a topic for extensive research due to both its mathematical beauty and practical applications in the field of facility location. The first algorithm for solving the Fermat–Torricelli problem was introduced in 1937 by Weiszfeld in [1]. This algorithm was studied in depth by Kuhn in [2]. The Fermat–Torricelli problem and Weiszfeld’s algorithm have been revisited and further studied by many authors; see, e.g., [3–5] and the references therein.

Several generalized models for the Fermat–Torricelli problem have been introduced and studied in the literature. The Fermat–Torricelli problem in general normed spaces was considered in [6]. The generalized Fermat–Torricelli problems involving Minkowski gauges and distances to convex sets were the topics of [7–10]. In particular, our recent paper [9] focused on numerical algorithms with the use of the Nesterov smoothing technique and accelerated gradient method to study these problems.

Given the locations of a finite number of “customers,” the multifacility location problem asks for the optimal locations of a finite number of “facilities” (also known as centroids) to serve these customers, where each customer is assigned to the nearest facility. The multifacility location problem has a close relationship to clustering problems. A recent paper by An, Belghiti, and Tao [11] uses the so-called difference of convex functions algorithm (DCA) to solve a clustering problem that involves squared Euclidean distances. Their method shows robustness, efficiency, and superiority compared with the well-known $K$-means algorithm, when applied to a number of real-world data sets. The DCA was introduced by Tao in 1986 and then extensively developed in the works of An, Tao, and others; see [12,13] and the references therein. An important feature of the DCA is its simplicity compared with other methods, while still being very effective for many applications. In fact, the DCA is one of the most successful algorithms for solving nonconvex optimization problems.

In this paper, we consider the weighted Fermat–Torricelli problem with both positive and negative weights. Additionally, we consider a continuous multifacility location problem, which involves distance measurements generated by Minkowski gauges. Considering Minkowski gauges, it is possible to unify the problems generated by arbitrary norms and even more generalized notions of distances; see [7–9] and the references therein. Our approach is based on the Nesterov smoothing technique [14] and the DCA. We also propose a method to solve a new model of clustering called set clustering. This model involves squared Euclidean distances to convex sets and hence coincides with the model considered in [11] when the sets reduce to singletons. Using sets, instead of points, allows us to classify objects with non-negligible sizes.

The remainder of the paper is organized as follows. In Sect. 2, we give an accessible presentation of DC programming and the DCA. Section 3 is devoted to developing algorithms to solve generalized weighted Fermat–Torricelli problems involving possibly negative weights and Minkowski gauges. Algorithms for solving multifacility location problems with Minkowski gauges are presented in Sect. 4. In Sect. 5, we introduce and develop an algorithm to solve the set clustering model. Finally, we demonstrate the effectiveness of our algorithms through a variety of numerical examples in Sect. 6 and give concluding remarks in Sect. 7.
2 Tools of Optimization

In this section, we provide basic results of DC programming and the DCA for the convenience of the reader. Most of the results in this section can be found in [12,13], although our presentation is tailored to the algorithms we present in the subsequent sections.

Consider the problem

\[
\text{minimize } f(x) := g(x) - h(x), \quad x \in \mathbb{R}^n.
\] (1)

where \( g: \mathbb{R}^n \to ]-\infty, +\infty[ \) and \( h: \mathbb{R}^n \to \mathbb{R} \) are convex functions. The function \( f \) in (1) is called a DC function and \( g - h \) is called a DC decomposition of \( f \).

For a convex function \( \varphi: \mathbb{R}^n \to ]-\infty, +\infty[ \), the Fenchel conjugate of \( \varphi \) is defined by

\[
\varphi^*(y) := \sup \{ \langle y, x \rangle - \varphi(x) : x \in \mathbb{R}^n \}.
\]

Note that if \( \varphi \) is proper, i.e., \( \text{dom}(\varphi) := \{ x \in \mathbb{R}^n : \varphi(x) < +\infty \} \neq \emptyset \), then \( \varphi^*: \mathbb{R}^n \to ]-\infty, +\infty[ \) is also a convex function. Given \( \bar{x} \in \text{dom}(\varphi) \), an element \( v \in \mathbb{R}^n \) is called a subgradient of \( \varphi \) at \( \bar{x} \) if

\[
\langle v, x - \bar{x} \rangle \leq \varphi(x) - \varphi(\bar{x}) \quad \text{for all } x \in \mathbb{R}^n.
\]

The collection of all subgradients of \( \varphi \) at \( \bar{x} \) is called the subdifferential of \( \varphi \) at this point and is denoted by \( \partial \varphi(\bar{x}) \). If \( g \) is proper and lower semicontinuous, then \( v \in \partial \varphi^*(y) \) if and only if \( y \in \partial \varphi(v) \); see, e.g., [15–17].

Introduced by Tao and An [12,13], the DCA is a simple but effective optimization scheme for minimizing differences of convex functions. Although the algorithm is used for nonconvex optimization problems, the convexity of the functions involved still plays a crucial role with the presence of elements of convex analysis such as subgradients and Fenchel conjugates. The algorithm is summarized below, as applied to (1).

**Algorithm 1.**

```
INPUT: \( x_1 \in \mathbb{R}^n, N \in \mathbb{N} \).
for \( k = 1, \ldots, N \) do
    Find \( y_k \in \partial h(x_k) \).
    Find \( x_{k+1} \in \partial g^*(y_k) \).
end for
OUTPUT: \( x_{N+1} \).
```

Let \( g, h: \mathbb{R}^n \to ]-\infty, +\infty[ \) be proper, lower semicontinuous, and convex functions. It is well known that \( v \in \partial g^*(y) \) if and only if

\[
v \in \text{argmin} \{ g(x) - \langle y, x \rangle : x \in \mathbb{R}^n \}.
\] (2)

Moreover, \( w \in \partial h(x) \) if and only if

\[
w \in \text{argmin} \{ h^*(y) - \langle y, x \rangle : y \in \mathbb{R}^n \}.
\] (3)
Thus, in the case where we cannot find $x_k$ or $y_k$ exactly in Algorithm 1, we can find them approximately by solving two convex optimization problems in each iteration, as in the algorithm below.

Algorithm 2.

INPUT: $x_1 \in \mathbb{R}^n, N \in \mathbb{N}$

for $k = 1, \ldots, N$ do

Find $y_k \in \partial h(x_k)$ or find $y_k$ approximately by solving the problem:

\[
\text{minimize } \psi_k(y) := h^*(y) - \langle x_k, y \rangle, \quad y \in \mathbb{R}^n.
\]

Find $x_{k+1} \in \partial g^*(y_k)$ or find $x_{k+1}$ approximately by solving the problem:

\[
\text{minimize } \phi_k(x) := g(x) - \langle x, y_k \rangle, \quad x \in \mathbb{R}^n.
\]

end for

OUTPUT: $x_{N+1}$

Let us now discuss the convergence of the DCA. Recall that a function $h: \mathbb{R}^n \to (-\infty, +\infty]$ is called $\gamma$-convex ($\gamma \geq 0$) if the function defined by $k(x) := h(x) - \frac{\gamma}{2} \|x\|^2$, $x \in \mathbb{R}^n$, is convex. If there exists $\gamma > 0$ such that $h$ is $\gamma-$convex, then $h$ is called strongly convex.

We say that an element $\bar{x} \in \mathbb{R}^n$ is a critical point of the function $f$ defined in (1) if

$$\partial g(\bar{x}) \cap \partial h(\bar{x}) \neq \emptyset.$$ 

Obviously, in the case where both $g$ and $h$ are differentiable, $\bar{x}$ is a critical point of $f$ if and only if $\bar{x}$ satisfies the Fermat rule $\nabla f(\bar{x}) = 0$.

The theorem below provides a convergence result for the DCA. The result can be derived directly from [13, Theorem 3.7].

**Theorem 2.1** Consider the function $f$ defined in (1) and the sequence $\{x_k\}$ generated by Algorithm 1. Then the following properties are valid:

(i) If $g$ is $\gamma_1$-convex and $h$ is $\gamma_2$-convex, then

$$f(x_k) - f(x_{k+1}) \geq \frac{\gamma_1 + \gamma_2}{2} \|x_{k+1} - x_k\|^2 \quad \text{for all } k \in \mathbb{N}. \quad (4)$$

(ii) The sequence $\{f(x_k)\}$ is monotone decreasing.

(iii) If $f$ is bounded from below, $g$ is lower semicontinuous, $g$ is $\gamma_1$-convex and $h$ is $\gamma_2$-convex with $\gamma_1 + \gamma_2 > 0$, and $\{x_k\}$ is bounded, then every subsequential limit of the sequence $\{x_k\}$ is a critical point of $f$.

In what follows, we discuss sufficient conditions for the constructibility of the sequence $\{x_k\}$, which give sufficient conditions for [13, Lemma 3.6].

**Proposition 2.1** Let $g:\mathbb{R}^n \to (-\infty, +\infty]$ be a proper, lower semicontinuous, and convex function. Then

$$\partial g(\mathbb{R}^n) := \bigcup_{x \in \mathbb{R}^n} \partial g(x) = \text{dom} \partial g^* := \{y \in \mathbb{R}^n : \partial g^*(y) \neq \emptyset\}.$$
Proof Let \( x \in \mathbb{R}^n \) and \( y \in \partial g(x) \). Then \( x \in \partial g^*(y) \), which implies \( \partial g^*(y) \neq \emptyset \), and so \( y \in \text{dom} \ \partial g^* \). The opposite inclusion is just as obvious. \( \square \)

We say that a function \( g : \mathbb{R}^n \to ] - \infty, +\infty] \) is coercive of superior order if

\[
\lim_{\|x\| \to +\infty} \frac{g(x)}{\|x\|} = +\infty.
\]

**Proposition 2.2** Let \( g : \mathbb{R}^n \to ] - \infty, +\infty] \) be a proper, coercive of superior order, and convex function. Then \( \text{dom} (\partial g^*) = \mathbb{R}^n \).

**Proof** By [15, Proposition 1.3.8] and the fact that \( f \) is proper, the Fenchel conjugate \( g^* \) is a finite convex function. Therefore, \( \partial g^*(y) \) is nonempty for all \( y \in \mathbb{R}^n \), which completes the proof. \( \square \)

### 3 The DCA for a Generalized Fermat–Torricelli Problem

In this section, we develop algorithms for solving weighted Fermat–Torricelli problems involving Minkowski gauges. In particular, the algorithms developed here are applicable to solving the unweighted version introduced and studied in [7]. Our method is based on the Nesterov smoothing technique and the DCA. This approach allows us to solve generalized versions of the Fermat–Torricelli problem generated by different norms and generalized distances.

Let \( F \) be a nonempty, closed, bounded, and convex set in \( \mathbb{R}^n \) containing the origin in its interior. Define the **Minkowski gauge** associated with \( F \) by

\[
\rho_F(x) := \inf \{ t > 0 : x \in tF \}.
\]

Note that if \( F \) is the closed unit ball in \( \mathbb{R}^n \), then \( \rho_F(x) = \|x\| \).

Given a nonempty and bounded set \( K \), the **support function** associated with \( K \) is given by

\[
\sigma_K(x) := \sup \{ \langle x, y \rangle : y \in K \}.
\]

It follows from the definition of the Minkowski function (see, e.g., [18, Proposition 2.1]) that \( \rho_F(x) = \sigma_{F^\circ}(x) \), where

\[
F^\circ := \{ y \in \mathbb{R}^n : \langle x, y \rangle \leq 1 \text{ for all } x \in F \}.
\]

Let us present below a direct consequence of the Nesterov smoothing technique given in [14]. In the proposition below, \( d(x; \Omega) \) denotes the Euclidean distance and \( P(x; \Omega) \) denotes the Euclidean projection from a point \( x \) to a nonempty, closed, and convex set \( \Omega \) in \( \mathbb{R}^n \).

**Proposition 3.1** Given any \( a \in \mathbb{R}^n \) and \( \mu > 0 \), a Nesterov smoothing approximation of \( \varphi(x) := \rho_F(x - a) \) has the representation

\[
\varphi_\mu(x) = \frac{1}{2\mu} \|x - a\|^2 - \frac{\mu}{2} \left[ d \left( \frac{x - a}{\mu} ; F^\circ \right) \right]^2.
\]
Moreover, \( \nabla \varphi_\mu(x) = P(\frac{x-a}{\mu}; F^\circ) \) and
\[
\varphi_\mu(x) \leq \varphi(x) \leq \varphi_\mu(x) + \frac{\mu}{2} \| F^\circ \|^2, \tag{5}
\]
where \( \| F^\circ \| := \sup\{\| u \| : u \in F^\circ\} \).

**Proof** The function \( \varphi \) can be represented as
\[
\varphi(x) = \sigma_{F^\circ}(x - a) = \sup\{\langle x - a, u \rangle : u \in F^\circ\}.
\]

Using the prox-function \( d(x) = \frac{1}{2} \| x \|^2 \) in [14], one obtains a smooth approximation of \( \varphi \) given by
\[
\varphi_\mu(x) := \sup\left\{ \langle x - a, u \rangle - \frac{\mu}{2} \| u \|^2 : u \in F^\circ \right\}
= \sup\left\{ -\frac{\mu}{2} \left( \| u \|^2 - \frac{2}{\mu} \langle x - a, u \rangle \right) : u \in F^\circ \right\}
= \sup\left\{ -\frac{\mu}{2} \| u - \frac{1}{\mu}(x - a) \|^2 + \frac{1}{2\mu} \| x - a \|^2 : u \in F^\circ \right\}
= \frac{1}{2\mu} \| x - a \|^2 - \frac{\mu}{2} \inf\left\{ \| u - \frac{1}{\mu}(x - a) \|^2 : u \in F^\circ \right\}
= \frac{1}{2\mu} \| x - a \|^2 - \frac{\mu}{2} \left[ d \left( \frac{x - a}{\mu}; F^\circ \right) \right]^2.
\]

The formula for computing the gradient of \( \varphi_\mu \) follows from the well-known gradient formulas for the squared Euclidean norm and the squared distance function generated by a nonempty, closed, and convex set: \( \nabla d^2(x; \Omega) = 2[x - P(x; \Omega)] \); see, e.g., [16, Exercise 3.2]. The estimate (5) can be proved directly. \( \Box \)

Let \( a^i \in \mathbb{R}^n \) for \( i = 1, \ldots, m \) and let \( c_i \neq 0 \) for \( i = 1, \ldots, m \) be real numbers. For the remainder of this section, we study the following generalized version of the Fermat–Torricelli problem:
\[
\text{minimize } f(x) := \sum_{i=1}^{m} c_i \rho_F \left( x - a^i \right), \ x \in \mathbb{R}^n. \tag{6}
\]

The function \( f \) in (6) can be written as
\[
f(x) = \sum_{c_i>0} c_i \rho_F \left( x - a^i \right) - \sum_{c_i<0} (-c_i) \rho_F \left( x - a^i \right).
\]

Let \( I := \{i : c_i > 0\} \) and \( J := \{i : c_i < 0\} \) with \( \alpha_i = c_i \) if \( i \in I \), and \( \beta_i = -c_i \) if \( i \in J \). Then
\[
f(x) = \sum_{i \in I} \alpha_i \rho_F \left( x - a^i \right) - \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right). \tag{7}
\]

\( \copyright \) Springer
An important step in applying Algorithm 1 for minimizing a function \( f \) represented as the difference of two convex functions \( g \) and \( h \) is to find subgradients of \( g^* \). The function \( f \) given in (7) has an obvious DC decomposition \( f = g - h \), where

\[
g(x) := \sum_{i \in I} \alpha_i \rho_F \left( x - a^i \right) \quad \text{and} \quad h(x) := \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right).
\]

However, there is no explicit formula for subgradients of \( \partial g^* \), and hence, we cannot apply Algorithm 1. Proposition 3.2 gives a Nesterov-type approximation for the function \( f \), which is favorable for applying this algorithm.

**Proposition 3.2** Consider the function \( f \) defined in (7). Given any \( \mu > 0 \), an approximation of the function \( f \) has the following DC decomposition:

\[
f_\mu(x) := g_\mu(x) - h_\mu(x), \ x \in \mathbb{R}^n,
\]

where

\[
g_\mu(x) := \sum_{i \in I} \frac{\alpha_i}{2\mu} \| x - a^i \|^2,
\]

\[
h_\mu(x) := \sum_{i \in I} \frac{\mu \alpha_i}{2} \left[ d \left( \frac{x - a^i}{\mu}; F^o \right) \right]^2 + \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right).
\]

Moreover, \( f_\mu(x) \leq f(x) \leq f_\mu(x) + \frac{\mu \| F^o \|^2}{2} \sum_{i \in I} \alpha_i \) for all \( x \in \mathbb{R}^n \).

**Proof** By Proposition 3.1,

\[
f_\mu(x) = \sum_{i \in I} \left[ \frac{\alpha_i}{2\mu} \| x - a^i \|^2 - \frac{\mu \alpha_i}{2} \left[ d \left( \frac{x - a^i}{\mu}; F^o \right) \right]^2 \right] - \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right)
\]

\[
= \sum_{i \in I} \frac{\alpha_i}{2\mu} \| x - a^i \|^2 - \left[ \sum_{i \in I} \frac{\mu \alpha_i}{2} \left[ d \left( \frac{x - a^i}{\mu}; F^o \right) \right]^2 \right] + \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right).
\]

The rest of the proof is straightforward. \( \Box \)

**Proposition 3.3** Let \( \gamma_1 := \sup \{ r > 0 : B(0; r) \subset F \} \) and \( \gamma_2 := \inf \{ r > 0 : F \subset B(0; r) \} \). Suppose that

\[
\gamma_1 \sum_{i \in I} \alpha_i > \gamma_2 \sum_{j \in J} \beta_j.
\]

Then the function \( f \) defined in (7) and its approximation \( f_\mu \) defined in Proposition 3.2 have absolute minima.

**Proof** Fix any \( r > 0 \) such that \( B(0; r) \subset F \). By the definition, for any \( x \in \mathbb{R}^n \),

\[
\rho_F(x) = \inf \left\{ t > 0 : t^{-1}x \in F \right\} \leq \inf \left\{ t > 0 : t^{-1}x \in B(0; r) \right\}
\]
This implies \( \rho_F(x) \leq \gamma_1^{-1}\|x\| \). Similarly, \( \rho_F(x) \geq \gamma_2^{-1}\|x\| \).

Then
\[
\sum_{i \in I} \alpha_i \rho_F \left( x - a^i \right) \geq \gamma_2^{-1} \sum_{i \in I} \alpha_i \left\| x - a^i \right\| \geq \gamma_2^{-1} \sum_{i \in I} \alpha_i \left( \|x\| - \|a^i\| \right),
\]
\[
\sum_{j \in J} \beta_j \rho_F \left( x - a^j \right) \leq \gamma_1^{-1} \sum_{j \in J} \beta_j \left( \|x\| + \|a^j\| \right).
\]

It follows that
\[
f(x) \geq \left( (\gamma_2)^{-1} \sum_{i \in I} \alpha_i - (\gamma_1)^{-1} \sum_{j \in J} \beta_j \right) \|x\| - c,
\]
where \( c := \gamma_2^{-1} \sum_{i \in I} \alpha_i \|a^i\| + \gamma_1^{-1} \sum_{j \in J} \beta_j \|a^j\| \).

The assumption guarantees that \( \lim_{\|x\| \to +\infty} f(x) = +\infty \), and so \( f \) has an absolute minimum.

By Proposition 3.2,
\[
f(x) \leq f_\mu(x) + \frac{\mu \|F\|}{2} \sum_{i \in I} \alpha_i.
\]

This implies that \( \lim_{\|x\| \to +\infty} f_\mu(x) = +\infty \), and so \( f_\mu \) has an absolute minimum as well. \( \square \)

Define
\[
h^1_\mu(x) := \sum_{i \in I} \frac{\mu \alpha_i}{2} \left[ d \left( \frac{x - a^i}{\mu}; F^\circ \right) \right]^2, \quad h^2_\mu(x) := \sum_{j \in J} \beta_j \rho_F \left( x - a^j \right).
\]

Then \( h_\mu = h^1_\mu + h^2_\mu \) and \( h^1_\mu \) is differentiable with
\[
\nabla h^1_\mu(x) = \sum_{i \in I} \alpha_i \left[ \frac{x - a^i}{\mu} - P \left( \frac{x - a^i}{\mu}; F^\circ \right) \right].
\]

**Proposition 3.4** Consider the function \( g_\mu \) defined in Proposition 3.2. For any \( y \in \mathbb{R}^n \), the function
\[
\phi_\mu(x) := g_\mu(x) - \langle y, x \rangle, x \in \mathbb{R}^n,
\]
has a unique minimizer given by
\[
x = \frac{y + \sum_{i \in I} \alpha_i a^i / \mu}{\sum_{i \in I} \alpha_i / \mu}.
\]
Proof  The gradient of the convex function $\phi_{\mu}$ is given by
\[
\nabla \phi_{\mu}(x) = \sum_{i \in I} \frac{\alpha_i}{\mu} (x - a_i) - y.
\]
The result then follows by solving $\nabla \phi_{\mu}(x) = 0$. 

Based on Algorithm 1, we present the algorithm below to solve the generalized Fermat–Torricelli problem (6):

Algorithm 3.

INPUT: $\mu > 0$, $x_1 \in \mathbb{R}^n$, $N \in \mathbb{N}$, $F$, $a^1$, $\ldots$, $a^m \in \mathbb{R}^n$, $c_1$, $\ldots$, $c_m \in \mathbb{R}$.

for $k = 1, \ldots, N$ do

Find $y_k = u_k + v_k$, where
\[
\begin{align*}
  u_k &:= \sum_{i \in I} \alpha_i \left[ \frac{x_k - a^i}{\mu} - P \left( \frac{x_k - a^i}{\mu}; F^\circ \right) \right], \\
v_k &\in \sum_{j \in J} \beta_j \partial \rho_F(x_k - a^j).
\end{align*}
\]
Find $x_{k+1} = \frac{y_k + \sum_{i \in I} \alpha_i a^i / \mu}{\sum_{i \in I} \alpha_i / \mu}$.

OUTPUT: $x_{N+1}$

Let us introduce another algorithm to solve the problem. This algorithm is obtained by using the Nesterov smoothing method for all functions involved in the problem and the following proposition. The proof of the proposition follows directly from Proposition 3.1, as in the proof of Proposition 3.2.

Proposition 3.5 Consider the function $f$ defined in (7). Given any $\mu > 0$, a smooth approximation of the function $f$ has the following DC decomposition:
\[
f_{\mu}(x) := g_{\mu}(x) - h_{\mu}(x), \quad x \in \mathbb{R}^n,
\]
where
\[
\begin{align*}
g_{\mu}(x) &:= \sum_{i \in I} \frac{\alpha_i}{2\mu} \|x - a^i\|^2, \\
h_{\mu}(x) &:= \sum_{j \in J} \frac{\beta_j}{2\mu} \|x - a^j\|^2 - \sum_{j \in J} \frac{\mu \beta_j}{2} \left[ d \left( \frac{x - a^j}{\mu}; F^\circ \right) \right]^2 \\
&\quad + \sum_{i \in I} \frac{\mu \alpha_i}{2} \left[ d \left( \frac{x - a^i}{\mu}; F^\circ \right) \right]^2.
\end{align*}
\]
Moreover,
\[
f_{\mu}(x) - \frac{\mu \|F^\circ\|^2}{2} \sum_{i \in I} \beta_i \leq f(x) \leq f_{\mu}(x) + \frac{\mu \|F^\circ\|^2}{2} \sum_{i \in I} \alpha_i
\]
for all $x \in \mathbb{R}^n$. 

\(\text{Springer}\)
Note that both functions $g_\mu$ and $h_\mu$ in Proposition 3.5 are smooth with the gradients given by

$$\nabla g_\mu(x) = \sum_{i \in I} \frac{\alpha_i}{\mu} (x - a^i),$$

$$\nabla h_\mu(x) = \sum_{j \in J} \frac{\beta_j}{\mu} (x - a^j) - \sum_{j \in J} \beta_j \left[ \frac{x - a_j}{\mu} - P \left( \frac{x - a_j}{\mu}; F^\circ \right) \right] + \sum_{i \in I} \alpha_i \left[ \frac{x - a_i}{\mu} - P \left( \frac{x - a_i}{\mu}; F^\circ \right) \right].$$

Based on this result and Algorithm 1, we obtain another algorithm for solving problem (6).

Algorithm 4.

**INPUT:** $\mu > 0$, $x_1 \in \mathbb{R}^n$, $N \in \mathbb{N}$, $F, a^1, \ldots, a^n \in \mathbb{R}^n$, $c_1, \ldots, c_m \in \mathbb{R}$.

for $k = 1, \ldots, N$ do

Find $y_k = u_k + v_k$, where

$$u_k := \sum_{i \in I} \alpha_i \left[ \frac{x_k - a^i}{\mu} - P \left( \frac{x_k - a^i}{\mu}; F^\circ \right) \right],$$

$$v_k := \sum_{j \in J} \beta_j \left[ P \left( \frac{x_k - a^j}{\mu}; F^\circ \right) \right].$$

Find $x_{k+1} = \frac{y_k + \sum_{i \in I} \alpha_i a^i / \mu}{\sum_{i \in I} \alpha_i / \mu}$.

**OUTPUT:** $x_{N+1}$

**Remark 3.1** When implementing Algorithm 3 and Algorithm 4, instead of using a fixed smoothing parameter $\mu$, we often change $\mu$ during the iteration. The general optimization scheme is

**Initialize:** $x_1 \in \mathbb{R}^n$, $\mu_0 > 0$, $\mu_* > 0$, $0 < \sigma < 1$.

Set $k = 1$

**Repeat the following**

Apply Algorithm 3 (or Algorithm 4) with $\mu = \mu_k$ and starting point $x_k$ to obtain an approximate solution $x_{k+1}$.

Update $\mu_{k+1} = \sigma \mu_k$.

**Until** $\mu_k \leq \mu_*$.

### 4 Multifacility Location

In this section, we consider multifacility location problems involving distances generated by Minkowski gauges. Given $a^i \in \mathbb{R}^n$ for $i = 1, \ldots, m$, we need to choose $x^\ell$
for \( \ell = 1, \ldots, k \) in \( \mathbb{R}^n \) as centroids and assign each member \( a^i \) to its closest centroid. The objective function to be minimized is the sum of the assignment distances:

\[
\text{minimize } f(x^1, \ldots, x^k) = \sum_{i=1}^{m} \min_{\ell=1, \ldots, k} \rho_F(x^\ell - a^i), \quad x^\ell \in \mathbb{R}^n, \ell = 1, \ldots, k. 
\]

(8)

Let us first discuss the existence of an optimal solution.

**Proposition 4.1** The optimization problem (8) admits a global optimal solution \((x^1, \ldots, x^k) \in (\mathbb{R}^n)^k\).

**Proof** We only need to consider the case where \( k < m \) because a global solution can be found by setting \( x^\ell = a^\ell \) for \( \ell = 1, \ldots, m \), and \( x^{\ell+1} = \cdots = x^k = a^m \). Choose \( r > 0 \) such that

\[
r > \max\{\rho_F(a^i) : i = 1, \ldots, m\} + \max\{\rho_F(a^i - a^j) : i \neq j\}.
\]

Define

\[
\Omega := \{(x^1, \ldots, x^k) \in (\mathbb{R}^n)^k : \rho_F(x^i) \leq r \text{ for all } i = 1, \ldots, k\}.
\]

Then \( \Omega \) is a compact set. It suffices to show that

\[
\inf\{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in \Omega\} = \inf\{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in (\mathbb{R}^n)^k\}.
\]

Fix any \((x^1, \ldots, x^k) \in (\mathbb{R}^n)^k\). Suppose without loss of generality that \( \rho_F(x^i) > r \) for all \( i = 1, \ldots, p \), where \( p \leq k \), and \( \rho_F(x^i) \leq r \) for all \( i = p + 1, \ldots, k \). Since \( \rho_F \) is subadditive,

\[
\rho_F(x^\ell - a^i) \geq \rho_F(x^\ell) - \rho_F(a^i) > r - \rho_F(a^i) \geq \rho_F(a^\ell - a^i)
\]

for all \( \ell = 1, \ldots, p \), \( i = 1, \ldots, m \).

Therefore,

\[
f\left(x^1, x^2, \ldots, x^k\right) = \sum_{i=1}^{m} \min_{\ell=1, \ldots, k} \rho_F(x^\ell - a^i)
\]

\[
\geq f\left(a^1, a^2, \ldots, a^p, x^{p+1}, \ldots, x^k\right)
\]

\[
\geq \inf\left\{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in \Omega\right\}.
\]

Thus, \( \inf\{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in \Omega\} \leq \inf\{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in (\mathbb{R}^n)^k\} \), which completes the proof. \( \square \)
For our DC decomposition, we start with the following formula:

\[
\min_{\ell=1,\ldots,k} \rho_F(x^\ell - a^i) = \sum_{\ell=1}^k \rho_F(x^\ell - a^i) - \max_{r=1,\ldots,k} \sum_{\ell=1, \ell \neq r}^k \rho_F(x^\ell - a^i).
\]

Then

\[
f(x^1, \ldots, x^k) = \sum_{i=1}^m \left[ \sum_{\ell=1}^k \rho_F(x^\ell - a^i) \right] - \sum_{i=1}^m \max_{r=1,\ldots,k} \sum_{\ell=1, \ell \neq r}^k \rho_F(x^\ell - a^i)\right].
\]

Similar to the situation with minimizing the function \( f \) in (7), this DC decomposition is not favorable for applying the DCA from Algorithm 1. Our approach here is to apply the Nesterov smoothing technique to obtain an approximation of the objective function favorable for applying the DCA.

By Proposition 3.1, the objective function \( f \) then has the following approximation:

\[
f_\mu(x^1, \ldots, x^k) = \frac{1}{2\mu} \sum_{i=1}^m \sum_{\ell=1}^k \|x^\ell - a^i\|^2 - \left[ \frac{\mu}{2} \sum_{i=1}^m \sum_{\ell=1}^k \left[ d \left( \frac{x^\ell - a^i}{\mu}; F^\circ \right) \right]^2 \right.
\]

\[+ \sum_{i=1}^m \max_{r=1,\ldots,k} \sum_{\ell=1, \ell \neq r}^k \rho_F(x^\ell - a^i)\right].
\]

Thus, \( f_\mu(x^1, \ldots, x^k) = g_\mu(x^1, \ldots, x^k) - h_\mu(x^1, \ldots, x^k) \) is a DC decomposition of the function \( f_\mu \), where \( g_\mu \) and \( h_\mu \) are convex functions defined by

\[
g_\mu(x^1, \ldots, x^k) := \frac{1}{2\mu} \sum_{i=1}^m \sum_{\ell=1}^k \|x^\ell - a^i\|^2 \quad \text{and} \quad h_\mu(x^1, \ldots, x^k) := \frac{\mu}{2} \sum_{i=1}^m \sum_{\ell=1}^k \left[ d \left( \frac{x^\ell - a^i}{\mu}; F^\circ \right) \right]^2 + \sum_{i=1}^m \max_{r=1,\ldots,k} \sum_{\ell=1, \ell \neq r}^k \rho_F(x^\ell - a^i).
\]

Let \( X \) be the \( k \times n \) matrix whose rows are \( x^1, \ldots, x^k \). We consider the inner product space \( \mathcal{M} \) of all \( k \times n \) matrices with the inner product of \( A, B \in \mathcal{M} \) given by

\[
\langle A, B \rangle := \text{trace}(AB^T) = \sum_{i=1}^k \sum_{j=1}^n a_{ij} b_{ij}.
\]

The norm induced by this inner product is the Frobenius norm.

Then define

\[
G_\mu(X) := g_\mu(x^1, \ldots, x^k) = \frac{1}{2\mu} \sum_{\ell=1}^k \sum_{i=1}^m (\|x^\ell\|^2 - 2\langle x^\ell, a^i \rangle + \|a^i\|^2)
\]

\[\text{Springer}\]
\begin{align*}
= \frac{1}{2\mu} (m\|X\|^2 - 2\langle X, B \rangle + k\|A\|^2) \\
= \frac{m}{2\mu} \|X\|^2 - \frac{1}{\mu} \langle X, B \rangle + \frac{k}{2\mu} \|A\|^2,
\end{align*}

where $A$ is the $m \times n$ matrix whose rows are $a^1, \ldots, a^m$ and $B$ is the $k \times n$ matrix with $a := \sum_{i=1}^m a^i$ for every row.

Then the function $G_\mu$ is differentiable with gradient given by

$$\nabla G_\mu(X) = \frac{m}{\mu} X - \frac{1}{\mu} B.
$$

From the relation $X = \nabla G^*_\mu(Y)$ if and only if $Y = \nabla G_\mu(X)$, one has

$$\nabla G^*_\mu(Y) = \frac{1}{m} (B + \mu Y).
$$

Let us now provide a formula to compute the subdifferential of $H_\mu$ (defined below) at $X$.

First, consider the function

$$H^1_\mu(X) := \frac{\mu}{2} \sum_{i=1}^m \sum_{\ell=1}^k \left[ d \left( \frac{x^\ell - a^i}{\mu}; F^\circ \right) \right]^2$$

\begin{align*}
= \frac{\mu}{2} \left\{ \left[ d \left( \frac{x^1 - a^1}{\mu}; F^\circ \right) \right]^2 + \cdots + \left[ d \left( \frac{x^1 - a^m}{\mu}; F^\circ \right) \right]^2 \right\} + \\
+ \frac{\mu}{2} \left\{ \left[ d \left( \frac{x^k - a^1}{\mu}; F^\circ \right) \right]^2 + \cdots + \left[ d \left( \frac{x^k - a^m}{\mu}; F^\circ \right) \right]^2 \right\}.
\end{align*}

The partial derivatives of $H^1_\mu$ are given by

\begin{align*}
\frac{\partial H^1_\mu}{\partial x^1}(X) &= \frac{x^1 - a^1}{\mu} - P \left( \frac{x^1 - a^1}{\mu}; F^\circ \right) + \cdots + \frac{x^1 - a^m}{\mu} - P \left( \frac{x^1 - a^m}{\mu}; F^\circ \right) \\
&= \sum_{i=1}^m \left[ \frac{x^1 - a^i}{\mu} - P \left( \frac{x^1 - a^i}{\mu}; F^\circ \right) \right], \\
\vdots \\
\frac{\partial H^1_\mu}{\partial x^k}(X) &= \frac{x^k - a^1}{\mu} - P \left( \frac{x^k - a^1}{\mu}; F^\circ \right) + \cdots + \frac{x^k - a^m}{\mu} - P \left( \frac{x^k - a^m}{\mu}; F^\circ \right) \\
&= \sum_{i=1}^m \left[ \frac{x^k - a^i}{\mu} - P \left( \frac{x^k - a^i}{\mu}; F^\circ \right) \right].
\end{align*}
The gradient $\nabla H_1^\mu(X)$ is the $k \times n$ matrix whose rows are $\frac{\partial H_1^\mu}{\partial x^i}(X)$, \ldots, $\frac{\partial H_1^\mu}{\partial x^k}(X)$.

Let $H_\mu(X) := h_\mu(x^1, \ldots, x^k)$. Then $H_\mu = H_1^\mu + H^2$, where

$$H^2(X) := \sum_{i=1}^m \max_{r=1, \ldots, k} \sum_{\ell=1, \ell \neq r}^k \rho_F \left( x^\ell - a^i \right).$$

In what follows, we provide a formula to find a subgradient of $H^2$ at $X$.

Define the function

$$F_{i,r}(X) := \sum_{\ell=1, \ell \neq r}^k \rho_F \left( x^\ell - a^i \right).$$

Choose the row vector $v_{i,\ell} \in \partial \rho_F(x^\ell - a^i)$ if $\ell \neq r$ and $v_{i,r} = 0$. Then the $k \times n$ matrix formed by the rows $v_{i,r}$ for $i = 1, \ldots, k$ is a subgradient of $F_{i,r}$ at $X$.

Define

$$F_i(X) := \max_{r=1, \ldots, k} F_{i,r}(X).$$

In order to find a subgradient of $F_i$ at $X$, we first find an index $r \in I_i(X)$, where

$$I_i(X) := \{ r = 1, \ldots, k : F_i(X) = F_{i,r}(X) \}.$$

Then, choose $V_i \in \partial F_{i,r}(X)$ and we have that $\sum_{i=1}^m V_i$ is a subgradient of the function $H^2$ at $X$. This results in our first algorithm for the multifacility location problem.

**Algorithm 5.**

**INPUT:** $X_1 \in \mathcal{M}$, $N \in \mathbb{N}$, $F$, $a^1, \ldots, a^m \in \mathbb{R}$.

for $k = 1, \ldots, N$ do

Find $Y_k = U_k + V_k$, where

$U_k := \nabla H_1^\mu(X_k) = V_k \in \partial H^2(X_k)$.

Find $X_{k+1} = \frac{1}{m}(B + \mu Y_k)$.

**OUTPUT:** $X_{N+1}$

Let us now present the second algorithm for solving the multifacility problem. By Proposition 3.1, the function $F_{i,r}(X) := \sum_{\ell=1, \ell \neq r}^k \rho_F(x^\ell - a^i)$ has the following smooth approximation:

$$F_{i,r}^{\mu}(X) = \sum_{\ell=1, \ell \neq r}^k \left[ \frac{1}{2\mu} \left\| x^\ell - a^i \right\|^2 - \frac{\mu}{2} \left[ d \left( \frac{x^\ell - a^i}{\mu}; F^o \right) \right]^2 \right].$$

For a fixed $r$, define the row vectors $v_{i,\ell} = P \left( \frac{x^\ell - a^i}{\mu}; F^o \right)$ if $\ell \neq r$ and $v_{i,r} = 0$. Then $\nabla F_{i,r}^{\mu}(X)$ is the $k \times n$ matrix $V_{i,r}$ formed by these rows.
Now we define the function \( F^{i}_{\mu}(X) := \max_{r=1, \ldots, k} F^{i,r}_{\mu}(X) \). This is an approximation of the function

\[
F^{i}(X) := \max_{r=1, \ldots, k} \sum_{\ell=1, \ell \neq r}^{k} \rho F\left(x^{\ell} - a^{i}\right).
\]

As a result, \( H^{2}_{\mu} := \sum_{i=1}^{m} F^{i}_{\mu} \) is an approximation of the function \( H^{2} \).

Define the active index set

\[
I^{i}_{\mu}(X) := \{r = 1, \ldots, k : F^{i}_{\mu}(X) = F^{i,r}_{\mu}(X)\}.
\]

Choose \( r \in I^{i}_{\mu}(X) \) and calculate \( V_{i} = \nabla F^{i,r}_{\mu}(X) \). Then \( V := \sum_{i=1}^{m} V_{i} \) is a subgradient of the function \( H^{2}_{\mu} \) at \( X \).

Algorithm 6.

```
INPUT: X_{1} \in \mathcal{M}, N \in \mathbb{N}, F, a^{1}, \ldots, a^{m} \in \mathbb{R}^{n}.

for k = 1, \ldots, N do
    Find Y_{k} = U_{k} + V_{k}, where
    \quad U_{k} := \nabla H^{1}_{\mu}(X_{k}), V_{k} \in \partial H^{2}_{\mu}(X_{k}).
    Find X_{k+1} = \frac{1}{m}(B + \mu Y_{k}).

OUTPUT: X_{N+1}
```

Remark 4.1 Similar to the case of Algorithm 3 and Algorithm 4, when implementing Algorithm 5 and Algorithm 6, instead of using a fixed smoothing parameter \( \mu \), we often change \( \mu \) during the iteration.

5 Set Clustering

In this section, we study the problem of set clustering, where the objects being classified are sets rather than points. Given a nonempty, closed, and convex set \( \Omega \subset \mathbb{R}^{n} \), observe that

\[
[d(x; \Omega)]^{2} = \inf \left\{ \|x - w\|^{2} : w \in \Omega \right\}
\]

\[
= \inf \left\{ \|x\|^{2} - 2\langle x, w \rangle + \|w\|^{2} : w \in \Omega \right\}
\]

\[
= \|x\|^{2} + \inf \left\{ \|w\|^{2} - 2\langle x, w \rangle : w \in \Omega \right\}
\]

\[
= \|x\|^{2} - \sup \left\{ \langle 2x, w \rangle - \|w\|^{2} : w \in \Omega \right\}.
\]

Proposition 5.1 Let \( \Omega \) be a nonempty, closed, and convex set in \( \mathbb{R}^{n} \). Define the function

\[
\varphi_{\Omega}(x) := \sup \left\{ \langle 2x, w \rangle - \|w\|^{2} : w \in \Omega \right\} = 2 \sup \left\{ \langle x, w \rangle - \frac{1}{2}\|w\|^{2} : w \in \Omega \right\}.
\]

\( \copyright \) Springer
Then \(\varphi\) is convex and differentiable with \(\nabla \varphi_{\Omega}(x) = 2P(x; \Omega)\).

**Proof** It follows from the representation of \([d(x; \Omega)]^2\) above that
\[
\varphi_{\Omega}(x) = \|x\|^2 - [d(x; \Omega)]^2.
\]

Note that the function \(\psi(x) := [d(x; \Omega)]^2\) is differentiable with \(\nabla \psi(x) = 2[x - P(x; \Omega)]\); see, e.g., [16, Exercise 3.2]. Then the function \(\varphi_{\Omega}\) is differentiable with
\[
\nabla \varphi_{\Omega}(x) = 2x - 2[x - P(x; \Omega)] = 2P(x; \Omega),
\]
which completes the proof. \(\square\)

Let \(\Omega^i\) for \(i = 1, \ldots, m\) be nonempty, closed, and convex sets in \(\mathbb{R}^n\). We need to choose \(x^\ell\) for \(\ell = 1, \ldots, k\) in \(\mathbb{R}^n\) as centroids and assign each member \(\Omega^i\) to its closest centroid. The objective function to be minimized is the sum of these distances.

Then we have to solve the optimization problem:

\[
\text{minimize } f(x^1, \ldots, x^k) := \sum_{i=1}^m \min_{\ell=1, \ldots, k} [d(x^\ell; \Omega^i)]^2, \quad x^\ell \in \mathbb{R}^n, \ell = 1, \ldots, k. \tag{9}
\]

**Proposition 5.2** Suppose that the convex sets \(\Omega_i\) for \(i = 1, \ldots, m\) are nonempty, closed, and bounded. Then (9) has a global optimal solution.

**Proof** Choose \(r > 0\) such that \(\Omega^i \subset B(0; r)\) for all \(i = 1, \ldots, m\). Fix \(a^i \in \Omega^i\) for \(i = 1, \ldots, m\). Define
\[
S := \{(x^1, \ldots, x^k) \in (\mathbb{R}^n)^k : \|x^i\| \leq 6r \text{ for } i = 1, \ldots, k\}.
\]

Let us show that
\[
\inf \{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in (\mathbb{R}^n)^k \} = \inf \{f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in S\}.
\]

Fix any \((x^1, \ldots, x^k) \in (\mathbb{R}^n)^k\). Without loss of generality, suppose that \(k < m\) and \(\|x^\ell\| > 6r\) for \(\ell = 1, \ldots, p\), and \(\|x^{p+1}\| \leq 6r, \ldots, \|x^k\| \leq 6r\), where \(p \leq k\). Let \(p^{\ell,i} := P(x^\ell; \Omega^i)\). Then for \(\ell = 1, \ldots, p\), we have
\[
[d(x^\ell; \Omega^i)]^2 = \|x^\ell - p^{\ell,i}\|^2
\]
\[
= \|x^\ell\|^2 - 2\langle x^\ell, p^{\ell,i} \rangle + \|p^{\ell,i}\|^2
\]
\[
\geq \|x^\ell\|^2 - 2\|x^\ell\|\|p^{\ell,i}\|
\]
\[
= \|x^\ell\|(|\|x^\ell\| - 2\|p^{\ell,i}\|) \geq \|x^\ell\|(6r - 2\|p^{\ell,i}\|) \geq 4r\|x^\ell\| \geq 4r^2.
\]

In addition, for all \(\ell = 1, \ldots, m\), we have
\[
[d(a^\ell; \Omega^i)]^2 \leq \|a^\ell - a^i\|^2 \leq 4r^2 \leq [d(x^\ell; \Omega^i)]^2.
\]
It follows that
\[
  f(x^1, \ldots, x^k) = \sum_{i=1}^{m} \min_{\ell=1, \ldots, k} [d(x^\ell; \Omega^i)]^2
\]
\[
  \geq f(a^1, \ldots, a^p, x^{p+1}, x^\ell)
\]
\[
  \geq \inf \{ f(x^1, \ldots, x^k) : (x^1, \ldots, x^k) \in S \}.
\]

The rest of the proof follows from the proof of Proposition 4.1. \qed

For our DC decomposition, we use the following formula
\[
  \min_{\ell=1, \ldots, k} [d(x^\ell; \Omega^i)]^2 = \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2 - \max_{\ell=1, \ldots, k, \ell \neq r} \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2.
\]

Then
\[
  f(x^1, \ldots, x^k) = \sum_{i=1}^{m} \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2 - \left[ \sum_{i=1}^{m} \max_{\ell=1, \ldots, k, \ell \neq r} \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2 \right]
\]
\[
  = \sum_{i=1}^{m} \sum_{\ell=1}^{k} \|x^\ell\|^2 - \left[ \sum_{i=1}^{m} \sum_{\ell=1}^{k} \varphi_{\Omega^i}(x^\ell) + \sum_{i=1}^{m} \max_{\ell=1, \ldots, k, \ell \neq r} \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2 \right].
\]

Define
\[
  g(x^1, \ldots, x^k) := \sum_{i=1}^{m} \sum_{\ell=1}^{k} \|x^\ell\|^2
\]
\[
  h(x^1, \ldots, x^k) := \sum_{i=1}^{m} \sum_{\ell=1}^{k} \varphi_{\Omega^i}(x^\ell) + \sum_{i=1}^{m} \max_{\ell=1, \ldots, k, \ell \neq r} \sum_{\ell=1}^{k} [d(x^\ell; \Omega^i)]^2.
\]

Then we have the DC decomposition \( f = g - h \).

For \( X \in \mathcal{M} \), define
\[
  G(X) := \sum_{i=1}^{m} \sum_{\ell=1}^{k} \|x^\ell\|^2 = m \|X\|^2.
\]

Thus, \( \nabla G^*(X) = \frac{1}{2m}(X) \).

Define
\[
  H^1(X) := \sum_{i=1}^{m} \sum_{\ell=1}^{k} \varphi_{\Omega^i}(x^\ell).
\]
Then

$$\frac{\partial H^1}{\partial x^1} = 2P(x^1; \Omega^1) + \cdots + 2P(x^1; \Omega^m),$$

$$\vdots$$

$$\frac{\partial H^1}{\partial x^k} = 2P(x^k; \Omega^1) + \cdots + 2P(x^k; \Omega^m).$$

Then $\nabla H^1(X)$ is the $k \times n$ matrix whose rows are $\frac{\partial H^1}{\partial x^i}$ for $i = 1, \ldots, k$.

Let us now present a formula to compute a subgradient of the function

$$H^2(X) = \sum_{i=1}^{m} \max_{r=1, \ldots, k} \sum_{\ell=1, \ell \neq r} \left[ d(x^\ell; \Omega^i) \right]^2.$$

Define

$$H^i_2(X) := \max_{r=1, \ldots, k} \sum_{\ell=1, \ell \neq r} \left[ d(x^\ell; \Omega^i) \right]^2 = \max_{r=1, \ldots, k} H^{i,r}_2,$$

where

$$H^{i,r}_2 := \sum_{\ell=1, \ell \neq r} \left[ d(x^\ell; \Omega^i) \right]^2.$$

Consider the following row vectors

$$v_{i,\ell} := 2(x^\ell - P(x^\ell; \Omega^i)) \text{ if } \ell \neq r,$$

$$v_{i,r} := 0.$$

Then $\nabla H^i_2$ is the $k \times n$ matrix whose rows are these vectors.

Define the active index set

$$I^i(X) := \{ r = 1, \ldots, k : H^{i,r}_2(X) = H^i_2(X) \}.$$

Choose $r \in I^i(X)$ and let $V_i := \nabla H^{i,r}_2(X)$. Then $V := \sum_{i=1}^{m} V_i$ is a subgradient of $H^2$ at $X$. This leads to our algorithm for solving the set clustering problem (9).
Algorithm 7.

INPUT: \( X \in \mathcal{M}, N \in \mathbb{N}, \Omega_1, \ldots, \Omega_m \in \mathbb{R}^n \).

for \( k = 1, \ldots, N \) do

Find \( Y_k = U_k + V_k \), where \( U_k := \nabla H_1(X_k) \), \( V_k \in \partial H_2(X_k) \).

Find \( X_{k+1} = \frac{1}{2m}(Y_k) \).

OUTPUT: \( X_N \).

6 Numerical Implementation

We demonstrate the above algorithms on several problems. All code is written in MATLAB and run on an Intel Core i5 3.00 GHz CPU with 8GB RAM. Unless otherwise stated, we use the closed Euclidean unit ball for the set \( F \) associated with the Minkowski gauge. In accordance with Remark 3.1, we use \( \mu^* = 10^{-6} \), decreasing \( \mu \) over 3 implementations, each of which runs until \( \sum_{k=1}^k d(x_j^k, x_{j-1}^k) < k \cdot 10^{-6} \), where \( k \) is the number of centers and \( j \) is the iteration counter. The starting value \( \mu_0 \) is specified in each example.

Example 6.1 In this example, we implement Algorithms 3 and 4 to solve a generalized Fermat–Torricelli problem with negative weights, as defined in (6). We choose \( m = 44 \) points \( a_i \) in \( \mathbb{R}^2 \) as follows. For \( i = 1, \ldots, 40 \), we choose distinct \( a_i \) from

\[
\{ C_h + (\cos(j\pi/5), \sin(j\pi/5)) : h = 1, \ldots, 4 ; \ j = 1, \ldots, 10 \}
\]

where each \( C_h \) is a distinct element in \( \{(\pm 5, \pm 5)\} \) for \( h = 1, \ldots, 4 \). For \( i = 41, \ldots, 44 \), we choose distinct \( a_i \in \{(0, 0), (1, 2), (-3, -1), (-2, 3)\} \). The weights are assigned \( c_i = 1 \) if \( 1 \leq i \leq 40 \) and \( c_i = -2 \) if \( 41 \leq i \leq 44 \). For the smoothing parameter, we use an initial \( \mu_0 = 0.1 \). Then Algorithms 3 and 4 converge to optimal solutions of \( x \approx (1.90, -2.00) \) using \( F \) as the closed Euclidean unit ball, and \( x \approx (4.19, -4.31) \) with \( F \) as the closed \( \ell_1 \) unit ball (Fig. 1).

Fig. 1 A generalized Fermat–Torricelli problem in \( \mathbb{R}^2 \). Each \( \times \) is a negatively weighted point; the optimal solution is represented by filled circle for the \( \ell_2 \) norm and open square for the \( \ell_1 \) norm.

© Springer
A generalized Fermat–Torricelli problem in $\mathbb{R}^2$. Each negative point has weight of $-1000$; each positive point has a weight of 1; the optimal solution is represented by filled circle for the $\ell_1$ norm.

Objective function values for Algorithm 4 for the generalized Fermat–Torricelli problem under the $\ell_1$ norm shown in Fig. 2.
Example 6.2 In this example, we implement Algorithms 3 and 4 to solve the generalized Fermat–Torricelli problem under the $\ell_1$ norm with randomly generated points as shown in Fig. 2. This synthetic data set has 10,000 points with weight $c_i = 1$ and three points with weight $c_i = -1000$. For the smoothing parameter, we use an initial $\mu_0 = 0.1$. Both algorithms converge to an optimal solution of $x \approx (17.29, 122.46)$. The convergence rate is shown in Fig. 3.

Example 6.3 We implement Algorithm 5 to solve multifacility location problems given by function (8). We use the following six real data sets\(^1\): WINE contains 178 instances of $k = 3$ wine cultivars in $\mathbb{R}^{13}$. The classical IRIS data set contains 150 observations in $\mathbb{R}^4$, describing $k = 3$ varieties of Iris flower. The PIMA data set contains 768 observations, each with 8 features describing the medical history of adults of Pima American Indian heritage. IONOSPHERE contains data on 351 radar observations in $\mathbb{R}^{34}$ of free electrons in the ionosphere. USCity\(^2\) contains the latitude and longitude of 1217 US cities; we use $k = 3$ centroids (Fig. 4).

Reported values (Table 1) are as follows: $m$ is the number of points in the data set; $n$ is the dimension; $k$ is the number of centers; $\mu_0$ is the starting value for the smoothing parameter $\mu$, as discussed in Remark 3.1 (in each case, $\sigma$ is chosen so that $\mu$ decreases to $\mu_* \approx 3$ in three iterations); $Iter$ is the number of iterations until convergence; $CPU$ is the computation time in seconds; $Obj\; val$ is the final value of the true objective

---

1 Available at https://archive.ics.uci.edu/ml/datasets.html.
2 http://www.realestate3d.com/gps/uslatlongdegmin.htm.
Table 1 Results for Example 6.3, the performance of Algorithm 5 on real data sets

|     | m  | n  | k  | $\mu_0$ | Iter | CPU   | Objval          |
|-----|----|----|----|---------|------|-------|-----------------|
| WINE| 178| 13 | 3  | 10      | 690  | 1.86  | $1.62922 \times 10^4$ |
| IRIS| 150| 4  | 3  | 0.1     | 314  | 0.66  | 96.6565         |
| PIMA| 768| 8  | 2  | 10      | 267  | 2.22  | $4.75611 \times 10^4$ |
| IONOSPHERE| 351| 34 | 2  | 0.1     | 391  | 1.68  | $7.93712 \times 10^2$ |
| US city| 1217| 2  | 3  | 1       | 940  | 16.0  | $1.14211 \times 10^4$ |

$X = \begin{bmatrix}
36.2350^\circ N & 77.7130^\circ W \\
41.1278^\circ N & 86.1934^\circ W \\
34.2681^\circ N & 95.3486^\circ W \\
35.1042^\circ N & 108.1652^\circ W \\
38.2494^\circ N & 120.1098^\circ W
\end{bmatrix}$

Fig. 5 Fifty most populous US cities, approximated by a ball proportional to their area. Each city is assigned to the closest of five centroids (filled circle), which are the optimal facilities. See Example 4

function (8), not the smoothed version $f_\mu$. Implementations of Algorithm 6 produced nearly identical results on each example and thus are not reported.

Example 6.4 We now use Algorithm 7 to solve a multifacility location problem involving distances to sets, rather than points. We consider the latitude and longitude of the 50 most populous US cities on a plate carrée projection. For demonstration purposes, we represent each city with a ball of radius $r = 0.1\sqrt{A/\pi}$, where $A$ is the city’s reported area in square miles. Coordinates and area for each city were taken from 2014 United
States Census Bureau data. \(^3\) Then Algorithm 7 is implemented to minimize function (9) with \(k = 5\) centroids. An optimal solution is given below, and shown in Fig. 5.

\[
X = \begin{bmatrix}
36.2350^\circ \text{N} & 77.7130^\circ \text{W} \\
41.1278^\circ \text{N} & 86.1934^\circ \text{W} \\
34.2681^\circ \text{N} & 95.3486^\circ \text{W} \\
35.1042^\circ \text{N} & 108.1652^\circ \text{W} \\
38.2494^\circ \text{N} & 120.1098^\circ \text{W}
\end{bmatrix}
\]

\section*{7 Conclusions}

Based on the DCA and the Nesterov smoothing technique, we develop algorithms to solve a number of continuous optimization problems of facility location. Our development continues the works in [11]. Although unconstrained optimization problems are considered, an easy technique using the \textit{indicator function} and the Euclidean projection would solve the constrained versions of the problems. Another important question is the convergence rate of the algorithms, which can be addressed using recent progress in applying the Kurdyka–Lojasiewicz inequality.

\section*{Acknowledgements}

The research of Nguyen Mau Nam was partially supported by the US National Science Foundation under Grant DMS-1411817.

\section*{References}

1. Weiszfeld, E.: Sur le point pour lequel la somme des distances de \(n\) points donnés est minimum. Tôhoku Math. J. \textbf{43}, 355–386 (1937)
2. Kuhn, H.W.: A note on Fermat–Torricelli problem. Math. Program. \textbf{4}, 98–107 (1973)
3. Brimberg, J.: The Fermat Weber location problem revisited. Math. Program. \textbf{71}, 71–76 (1995)
4. Drezner, Z.: On the convergence of the generalized Weiszfeld algorithm. Ann. Oper. Res. \textbf{167}, 327–336 (2009)
5. Kupitz, Y.S., Martini, H.: Geometric aspects of the generalized Fermat–Torricelli problem. Bolyai Soc. Math. Stud. \textbf{6}, 55–127 (1997)
6. Martini, H., Swanepoel, K.J., Weiss, G.: The Fermat–Torricelli problem in normed planes and spaces. J. Optim. Theory Appl. \textbf{115}, 283–314 (2002)
7. Jahn, T., Kupitz, Y.S., Martini, H., Richter, C.: Minsum location extended to gauges and to convex sets. J. Optim. Theory Appl. \textbf{166}, 711–746 (2015)
8. Mordukhovich, B.S., Nam, N.M.: Applications of variational analysis to a generalized Fermat–Torricelli problem. J. Optim. Theory Appl. \textbf{148}, 431–454 (2011)
9. Nam, N.M., An, N.T., Rector, R.B., Sun, J.: Nonsmooth algorithms and Nesterov’s smoothing technique for generalized Fermat–Torricelli problems. SIAM J. Optim. \textbf{24}, 1815–1839 (2014)
10. Nam, N.M., Hoang, N.: A generalized Sylvester problem and a generalized Fermat–Torricelli problem. J. Convex Anal. \textbf{20}, 669–687 (2013)
11. An, L.T.H., Belghiti, M.T., Tao, P.D.: A new efficient algorithm based on DC programming and DCA for clustering. J. Glob. Optim. \textbf{27}, 503–608 (2007)
12. Tao, P.D., An, L.T.H.: Convex analysis approach to D.C. programming: Theory, algorithms and applications. Acta Math. Vietnam. \textbf{22}, 289–355 (1997)
13. Tao, P.D., An, L.T.H.: A d.c. optimization algorithm for solving the trust-region subproblem. SIAM J. Optim. \textbf{8}, 476–505 (1998)

\(^3\) Available at https://en.wikipedia.org/wiki/List_of_United_States_cities_by_population.
14. Nesterov, Yu.: Smooth minimization of non-smooth functions. Math. Program. 103, 127–152 (2005)
15. Hiriart-Urruty, J.B., Lemaréchal, C.: Fundamentals of Convex Analysis. Springer-Verlag, Berlin (2001)
16. Mordukhovich, B.S., Nam, N.M.: An Easy Path to Convex Analysis and Applications. Morgan & Claypool Publishers, San Rafael (2014)
17. Rockafellar, R.: T: Convex Analysis. Princeton University Press, Princeton, NJ (1970)
18. He, Y., Ng, K.F.: Subdifferentials of a minimum time function in Banach spaces. J. Math. Anal. Appl. 321, 896–910 (2006)