The Optimal Solution Set of the Multi-source Weber Problem

A. Raeisi Dehkordi¹

Received: 2 July 2017 / Accepted: 19 April 2018 / Published online: 12 July 2018
© The Author(s) 2018

Abstract

This paper considers the classical multi-source Weber problem (MWP), which is to find M new facilities with respect to N customers to minimize the sum of transportation costs between these facilities and the customers. We propose a modified algorithm in the spirit of Cooper's work for solving the MWP including a location phase and an allocation phase. The task of location phase is to find the optimal solution sets of many single-source Weber problems (SWPs), which are reduced by the heuristic of the nearest center reclassification for the customers in the previous allocation phase. Some examples are stated to clarify the proposed algorithms. Moreover, we present an algorithm with $O(d \log d)$ time for finding the optimal solution set of SWP in the collinear case; where d is the number of customers.

Keywords Multi-source Weber problem · Location · Subdifferential

Mathematics Subject Classification Primary 90B06; Secondary 46N10 · 49J52

1 Introduction

The single-source Weber problem (SWP), also known as the Fermat–Weber location problem, which is probably the first facility location problem historically, was studied as early as the 17th century. This is the simplest continuous facility location model; however, there were still questions, which were resolved only very recently. The Fermat–Weber location problem intends to find a point in $\mathbb{R}^n$ that minimizes the sum of the weighted Euclidean distance from the d given points:

Communicated by Sohrab Effati.

✉ A. Raeisi Dehkordi
a.raisi@sci.ui.ac.ir

¹ Department of Mathematics, University of Isfahan, Esfahān, Iran

$$\min_{x \in \mathbb{R}^n} f(x) := \sum_{j=1}^d s_j \|x - a_j\|,$$

where $a_j \in \mathbb{R}^n$ ($j = 1, 2, \dots, d$) is the location of the j th customer, and $s_j > 0$ ($j = 1, 2, \dots, d$) is the weight corresponding to the customer a_j . It is well-known that if the data points are not collinear, f is a strictly convex function and therefore has a unique optimal solution. In the collinear case, SWP can be reduced to a selecting problem (see [2, 16, 21]). Among the several schemes to solve SWP [3, 7, 8, 10, 14, 18, 22, 28], one of the most popular methods was presented in [27], and has been discussed and developed widely in [1, 6, 12, 15, 17, 19, 26]. In [3] the objective function of SWP is approximated by a simple function, and an ϵ -approximation algorithm is presented for solving a new problem with this approximated function. In [18] the Newton-Bracketing (NB) method for convex minimization was utilized to solve the SWP. This paper presents a fast algorithm for solving SWP in the collinear case. The advantages of this algorithm are three folds:

1. It can find the optimal solution set (or the set of all optimal solutions) of SWP.
2. It is simple and easy to implement.
3. It requires little computational time.

There are several algorithms (see for instance [16]) to sort the points in $\mathbb{R}$ in nondecreasing order, to the best of our knowledge there exists no algorithm for sorting the points in $\mathbb{R}^n$, which are collinear. In our algorithm, for solving SWP, we will need to sort the points in $\mathbb{R}^n$. For this purpose, we present an algorithm in $O(d \log(d))$ time.

Here, we consider the classical multi-source Weber problem (MWP) which plays an important role in the operations research and management science. The classical MWP is to find the locations of M new facilities to minimize the sum of the transportation costs between these facilities, and the N customers with known locations. We assume that the involved transportation costs are proportional to the corresponding distances. More specifically, the mathematical model of MWP is as follows:

$$\begin{aligned} \text{MWP : } \min F(x_1, x_2, \dots, x_M) &:= \sum_{i=1}^M \sum_{j=1}^N w_{ij} \|x_i - a_j\|, \\ \sum_{i=1}^M w_{ij} &= s_j, \quad j = 1, 2, \dots, N, \\ x_i &\in \mathbb{R}^n, \quad i = 1, 2, \dots, M, \end{aligned} \quad (1.1)$$

where,

- (1) $a_j \in \mathbb{R}^n$ is the location of the j th customer, $j = 1, 2, \dots, N$;
- (2) $x_i \in \mathbb{R}^n$ is the location of the i th facility to be determined, $i = 1, 2, \dots, M$;
- (3) $s_j \geq 0$ is the given demand required by the j th customer;
- (4) $w_{ij} \geq 0$ denotes the unknown allocation from the i th facility to the j th customer;
and
- (5) $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^n$.

It is well-known that the objective function of the problem (1.1) is neither concave nor convex, and may have several local optimal solutions, see [9]. Furthermore, [25] demonstrates that the problem is NP-hard, even if all of the demand points are to be located on a straight line. Among the existing effective numerical algorithms for solving MWP is the Cooper's algorithm, which was presented originally in [9]. Its attractive characteristic is that each iteration consists of a location phase and an allocation phase. However, there are similar algorithms [14, 18], based on Cooper's algorithm for solving MWP. According to [9], the Cooper's algorithm has a monotone nonincreasing convergent sequence for the objective function value. Nevertheless, it will not guarantee to converge to the global minimum. As the Cooper's algorithm is swift, straightforward and efficient, it has widely been used by other researchers either to generate efficient initial solutions or improve the obtained results even further [4, 11, 20].

In this paper, we present a modified algorithm in the spirit of Cooper's work for solving MWP, whose important characteristics are that it considers the optimal solution set and improves the desirability of the solutions in the Cooper's algorithm. Since the facilities are capacitated, it can easily be proved that in an optimal solution of the MWP, each customer is satisfied by its nearest facility. In the location phase of modified algorithm, we find the optimal solution set instead of an optimal solution in Cooper's algorithm. Then, in the allocation phase of the modified algorithm, we peruse that each customer is assigned to its nearest facility among all optimal solutions in the location phase that it leads to find near solutions to the best solutions of MWP.

The paper is organized as follows. In the next section, we will discuss the Cooper's algorithm as it will be used as a basis in the design of our algorithm. Section 3 provides some preliminaries which will be used in the sequel. In Sect. 4, we consider the SWP in the collinear case and an algorithm presents for solving this problem. In Sect. 5, a modified Cooper's algorithm in the spirit of the Cooper's work is developed. In Sect. 6, an application for the importance of finding the optimal solution set of MWP is presented.

2 A Review of Literature

In [9] an iterative heuristic method known as the alternate location-allocation algorithm for solving MWP is presented. This algorithm is efficient in terms of solution quality and computational effort. The base of the Cooper's algorithms is to decrease the objective function value of MWP in each iteration. Let $\mathcal{N} = \{1, 2, \dots, N\}$ and $A = \{a_j \mid j \in \mathcal{N}\}$ denote the set of locations of all customers, and $A^k = \{A_1^k, A_2^k, \dots, A_M^k\}$ with $\bigcup_{i=1}^M A_i^k = A$ and $A_i^k \cap A_j^k = \emptyset$ (for $i \neq j$) denote the disjoint partition of A at the k th iteration. At the $(k + 1)$ th iteration of the Cooper's algorithm, the location phase finds the candidate of locations of facilities by solving the following SWPs:

$$\begin{aligned} \text{SWP} : x_i^{k+1} &\in \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ F_i(x) := \sum_{\{j \in \mathcal{N}, a_j \in A_i^k\}} s_j \|x - a_j\| \right\}, \\ &= \{y \in \mathbb{R}^n \mid F_i(y) \leq F_i(x) \quad \forall x \in \mathbb{R}^n\} \quad i = 1, 2, \dots, M, \end{aligned} \quad (2.1)$$

where, A_i^k for $i = 1, 2, \dots, M$ are produced by the allocation phase. Then, the allocation phase involves an allocation or a partition, which depends on the x_i^{k+1} ($i = 1, 2, \dots, M$) generated by solving (2.1). More specifically, if for all $i \in \{1, 2, \dots, M\}$, x_i^{k+1} is the nearest facility among all facilities for each customer in A_i^k , then A_i^k ($i = 1, 2, \dots, M$) are the desirable partitions. Therefore, it is reasonable to allocate customers from the facility x_i^{k+1} to minimize the total sum of transportation costs. Otherwise, the set of customers should be partitioned again according to the heuristic of the nearest center reclassification (*NCR*), i.e., the new partition $A = \{A_1^{k+1}, A_2^{k+1}, \dots, A_M^{k+1}\}$ is generated so that x_i^{k+1} is the nearest facility for each customer in A_i^{k+1} . Note that with the presupposition that each facility to be determined is capable of providing sufficient services for the targeted customers, the heuristic characteristic shared by all Cooper-type algorithms is that, finally, each customer (a_j) is served only by one of the facilities (the nearest one). This observation explains the disappearance of w_{ij} in (2.1). The MWP has considerably been probed in the literature as a widely used optimization problem. A branch and bound algorithm has been developed in [24] to solve the problem. The exact methods are not capable of dealing with large-sized instances in a reasonable computational time, hence heuristic methods appeared to be the best way forward. In [13], MWP is solved by a p -median problem, and then the Cooper's algorithm is applied to find the proper locations for the facilities. In [4] a solution approach is presented by proposing a combination of the variable neighborhood search and the Cooper's algorithm. A two-phase heuristic method is investigated in [11], known as a cellular heuristic.

3 Preliminaries

In this section, we present some notations and auxiliary results that will be needed in what follows. Let $a, b \in \mathbb{R}^n$. We denote the line segment between a and b by $[a, b]$; i.e.,

$$[a, b] = \{ta + (1 - t)b \mid 0 \leq t \leq 1\}.$$

The closed unit ball in $\mathbb{R}^n$ is denoted by $\mathbb{B}$. The optimal solution set of SWP is denoted by

$$\Omega := \{x^* \in \mathbb{R}^n \mid f(x^*) \leq f(x) \ \forall x \in \mathbb{R}^n\}.$$

Let $x \in \mathbb{R}$. We denote the largest integer less than or equal to x (or the floor function at x) by $\lceil x \rceil$. Let X be a non-empty closed set in $\mathbb{R}^n$. Let y be a point in X that is closest to x ; we say that y lies in the projection of x onto X . The set of points in X closest to x is denoted by

$$P_X(x) := \operatorname{argmin}_{s \in X} \|x - s\| = \{y \in X \mid \|x - y\| \leq \|x - s\| \ \forall s \in X\}.$$

We recall the following results from [23]. Let $\hat{x} \in \mathbb{R}^n$ and $\phi : \mathbb{R}^n \rightarrow \mathbb{R}$ be a convex and Lipschitz function. In place of the gradient, we consider subgradients, those elements ξ of $\mathbb{R}^n$ satisfying:

$$\phi(x) - \phi(\hat{x}) \geq \langle \xi, x - \hat{x} \rangle \quad \forall x \in \mathbb{R}^n.$$

The set of subgradients (called the subdifferential) is denoted by $\partial\phi(\hat{x})$.

In the following theorem, we summarize some results, which are used in what follows:

Theorem 3.1 [23] *Let $\phi : \mathbb{R}^n \rightarrow \mathbb{R}$ and $\psi : \mathbb{R}^n \rightarrow \mathbb{R}$ be Lipschitz and convex functions. Then, the following assertions hold:*

- (1) *For any scalar λ , $\partial\lambda\phi(x) = \lambda\partial\phi(x)$.*
- (2) *The point $\hat{x}$ is a (global) minimizer of ϕ , if and only if $0 \in \partial\phi(\hat{x})$.*
- (3) *Let ψ be differentiable at $\hat{x}$, then $0 \in \partial(\phi + \psi)(\hat{x})$, if only if:*

$$-\nabla\psi(\hat{x}) \in \partial\phi(\hat{x}).$$

- (4) $\partial(\|\cdot\| - a\|)(a) = \mathbb{B}$.

4 The Nonsmooth Approach to SWP

The objective function of SWP is a nonsmooth function with non-differentiable points $a_1, a_2, \dots, a_d$, which makes it hard to solve. This section considers the case that the data points are collinear. By following the well-known lemma, we are able to recognize the collinearity of the given points $a_1, a_2, \dots, a_d$.

Lemma 4.1 *Suppose that $a_2 - a_1, a_3 - a_2, \dots, a_d - a_1$ are the columns of the matrix $\mathcal{A}$. Then $a_1, a_2, \dots, a_d$ are collinear if and only if $\mathcal{A}$ has rank 1 or less.*

Definition 4.2 We say that the list $a_1, a_2, \dots, a_d$ is regular if $[a_i, a_j] \subseteq [a_p, a_q]$ for $1 \leq p \leq i \leq j \leq q \leq d$.

In the following theorem, we give a rearrangement of points to get a list of regular points. In this section, we suppose that $a_1, a_2, \dots, a_d$ are distinct and collinear.

Theorem 4.3 *The given points $a_1, a_2, \dots, a_d$ can be rearranged into $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$ so that there exists $k \in \{1, 2, \dots, n\}$ such that*

$$a_{k,\pi(1)} < a_{k,\pi(2)} < \dots < a_{k,\pi(d)},$$

where, $\pi : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\}$ is a permutation and $a_{k,\pi(j)}$ ($j = 1, 2, \dots, d$) is the k th coordinate of $a_{\pi(j)}$. Suppose that $a_1, a_2, \dots, a_d$ are rearranged as $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$, then $\{a_{k,\pi(j)}\}_{j=1}^d$ is a constant or strictly monotone sequence for each $k \in \{1, 2, \dots, n\}$. In particular, the list $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$ is regular.

Proof The proof is trivial. □

Let $\mathbb{A} = (a_1, a_2, \dots, a_d)$. To regularize $a_1, a_2, \dots, a_d$, Theorem 4.3 suggests a simple algorithm, which requires sorting the elements of one of the rows $\mathbb{A}$ as $a_{k,1}, a_{k,2}, \dots, a_{k,d}$ such that $a_{k,1} \neq a_{k,2}$.

Sort Algorithm

<i>Input:</i>	A list $a_1, a_2, \dots, a_d$ of customers.
<i>Output:</i>	A permutation $\pi : \{1, 2, \dots, d\} \rightarrow \{1, 2, \dots, d\}$ such that $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$ be regular.
<i>Step 1.</i>	Find $k \in \{1, 2, \dots, n\}$ such that $a_{k,1} \neq a_{k,2}$.
<i>Step 2.</i>	Sort $a_{k,1}, a_{k,2}, \dots, a_{k,d}$, i.e., find a permutation $\pi : \{1, 2, \dots, d\} \rightarrow \{1, 2, \dots, d\}$ such that $a_{k,\pi(i)} < a_{k,\pi(i+1)}$ for all $i = 1, 2, \dots, d-1$.
<i>Step 3.</i>	List $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$ is regular.

Note that the list $a_{k,1}, a_{k,2}, \dots, a_{k,d}$ in Step 2 can be sorted in $O(d \log d)$ time using the merge sort algorithm in [16]. Therefore, the overall running time (or the time complexity) of the sort algorithm is $O(n) + O(d \log d)$ time. By considering the sort algorithm, we assume that the list $a_1, a_2, \dots, a_d$ is regular.

In the following theorem, we discuss the convexity and closedness of the optimal solution set of SWP.

Theorem 4.4 *The set Ω is a closed and convex set. In particular, Ω is a subset of $[a_1, a_d]$.*

Proof It is easy to see that Ω is a closed and convex set. Now we denote by h the straight line passing through $a_1, a_2, \dots, a_d$,

$$h = \{a_1 + t(a_d - a_1), t \in \mathbb{R}\}.$$

Since the list $a_1, a_2, \dots, a_d$ is distinct and regular, there exists $k \in \{1, 2, \dots, n\}$ such that

$$a_{k,1} < a_{k,2} < \dots < a_{k,d}.$$

By contradiction, suppose that $x^* \in \Omega$ and $x^* \notin [a_1, a_d]$. If $x^* \notin h$, then $P_h(x^*) \neq x^*$ and

$$\|x^* - a_j\|^2 = \|x^* - P_h(x^*)\|^2 + \|P_h(x^*) - a_j\|^2, \quad j = 1, 2, \dots, d.$$

Therefore,

$$\|P_h(x^*) - a_j\| \leq \|x^* - a_j\|, \quad j = 1, 2, \dots, d,$$

with at least one strict inequality. Multiplying the above inequalities by s_j and adding them, we get

$$\sum_{j=1}^d s_j \|P_h(x^*) - a_j\| < \sum_{j=1}^d s_j \|x^* - a_j\|, \quad j = 1, 2, \dots, d.$$

This contradiction shows that x^* can not be an optimal solution for SWP. Now suppose that $x^* \in h$ and $x^* \notin [a_1, a_d]$. To proceed, consider the two possible cases: (a) $x_k^* > a_{k,d}$, and (b) $x_k^* < a_{k,1}$. In case (a), $P_{[a_1, a_d]}(x^*) = a_d$ and

$$\|x^* - a_j\| = \|x^* - a_d\| + \|a_d - a_j\|, \quad j = 1, 2, \dots, d-1.$$

Therefore,

$$\|a_d - a_j\| < \|x^* - a_j\|, \quad j = 1, 2, \dots, d-1.$$

Multiplying the above inequalities by s_j and adding them, we obtain

$$\sum_{j=1}^d s_j \|a_d - a_j\| < \sum_{j=1}^d s_j \|x^* - a_j\|, \quad j = 1, 2, \dots, d.$$

This is a contradiction. The case (b) can be done similarly to the case (a). □

In the following theorem, a necessary and sufficient condition for computing the optimal solution set of SWP is presented.

Theorem 4.5 *There exist $p, q \in \{1, 2, \dots, d\}$ such that $\Omega = [a_p, a_q]$.*

Proof According to Theorem 4.4, we have $\Omega = [x^*, y^*]$ where $x^*, y^* \in [a_1, a_d]$. On the contrary, suppose that $x^* \notin \{a_1, a_2, \dots, a_d\}$. Since x^* and $a_1, a_2, \dots, a_d$ are distinct and collinear, it follows from Theorem 4.3 that there exist $p \in \{1, 2, \dots, d\}$ and $k \in \{1, 2, \dots, n\}$ such that $a_{k,p} < x_k^* < a_{k,p+1}$. Since x^* is an optimal solution of SWP, we have $0 \in \partial f(x^*)$. By differentiability of f at x^* ,

$$\sum_{j=1}^d s_j \frac{x^* - a_j}{\|x^* - a_j\|} = 0.$$

We prove that $0 \in \partial f(a_p)$. For this purpose, we first show that

$$\frac{x^* - a_j}{\|x^* - a_j\|} = \frac{a_p - a_j}{\|a_p - a_j\|}, \quad j = 1, \dots, d, \quad j \neq p.$$

If $j \in \{1, \dots, p-1\}$, then there exists $t_j > 1$ such that $x^* = a_j + t_j(a_p - a_j)$. Therefore,

$$\frac{x^* - a_j}{\|x^* - a_j\|} = \frac{a_j + t_j(a_p - a_j) - a_j}{\|a_j + t_j(a_p - a_j) - a_j\|} = \frac{a_p - a_j}{\|a_p - a_j\|}, \quad j = 1, \dots, p-1.$$

If $j \in \{p+1, \dots, d\}$, then there exists $t_j \in (0, 1)$ such that $x^* = a_p + t_j(a_j - a_p)$. Therefore,

$$\frac{x^* - a_j}{\|x^* - a_j\|} = \frac{a_p + t_j(a_j - a_p) - a_j}{\|a_p + t_j(a_j - a_p) - a_j\|} = \frac{a_p - a_j}{\|a_p - a_j\|}, \quad j = p+1, \dots, d.$$

It implies that

$$\left\| -\sum_{\substack{j=1 \\ j \neq p}}^d s_j \frac{a_p - a_j}{\|a_p - a_j\|} \right\| = \left\| -\sum_{\substack{j=1 \\ j \neq p}}^d s_j \frac{x^* - a_j}{\|x^* - a_j\|} \right\| = \left\| s_p \frac{x^* - a_p}{\|x^* - a_p\|} \right\| = s_p.$$

Hence,

$$-\sum_{\substack{j=1 \\ j \neq p}}^d s_j \frac{a_p - a_j}{\|a_p - a_j\|} \in s_p \partial \|\cdot - a_p\|_{(a_p)}.$$

Therefore, $0 \in \partial f(a_p)$.

Since f is a convex function, it follows that $a_p \in \Omega$, which is a contradiction. Similarly, there exists $q \in \{1, 2, \dots, d\}$ such that $y^* = a_q$. $\square$

For convenience, we assume that $a_0, a_{d+1} \in \mathbb{R}^n$ such that $a_0, a_1, \dots, a_{d+1}$ are distinct and collinear, and also the list $a_0, a_1, \dots, a_{d+1}$ is regular. We set $s_0 := 0$, $s_{d+1} := 0$, $T_0 := \sum_{j=1}^d s_j$,

$$T_k := \sum_{j=k+1}^{d+1} s_j - \sum_{j=0}^{k-1} s_j, \quad k = 1, 2, \dots, d,$$

and $T_{d+1} := -\sum_{j=1}^d s_j$. Observe that

$$T_k = T_{k-1} - s_k - s_{k-1}, \quad k = 1, 2, \dots, d,$$

and

$$T_k = T_{k+1} + s_k + s_{k+1}, \quad k = 1, 2, \dots, d.$$

We propose the following algorithms for solving SWP in the collinear case.

Backward algorithm

<i>Input:</i>	The number of customers d and positive multipliers $s_1, s_2, \dots, s_d$.
<i>Output:</i>	The optimal solution set of SWP.
<i>Step 1.</i>	Pick an arbitrary $r \in \{2, 3, \dots, d, d+1\}$ such that $T_r < -s_r$.
<i>Step 2.</i>	Set $l = 0$.
<i>Step 3.</i>	For $k = r - 1$ to 1 do : $T_k = T_{k+1} + s_k + s_{k+1}$; if $ T_k \leq s_k$, then set $l = l + 1$ and $P(l) = k$; else, if $T_k > s_k$, then stop.
<i>Step 4.</i>	$[a_{P(1)}, a_{P(l)}]$ is the optimal solution set of SWP.

The backward algorithm can be replaced by the forward algorithm.

Forward algorithm

<i>Input:</i>	The number of customers d and positive multipliers $s_1, s_2, \dots, s_d$.
<i>Output:</i>	The optimal solution set of SWP.
<i>Step 1.</i>	Pick an arbitrary $r \in \{0, 1, \dots, d-1\}$ such that $T_r > s_r$.
<i>Step 2.</i>	Set $l = 0$.
<i>Step 3.</i>	For $k = r + 1$ to d do : $T_k = T_{k-1} - s_k - s_{k-1}$; if $ T_k \leq s_k$, then set $l = l + 1$ and $P(l) = k$; else, if $T_k < -s_k$, then stop.
<i>Step 4.</i>	$[a_{P(1)}, a_{P(l)}]$ is the optimal solution set of SWP.

To make these algorithms clear, the following example is provided.

Example 4.6 Consider six customers whose locations denoted by a_j ($j = 1, 2, \dots, 6$) are given by the columns of the following matrix:

$$\mathbb{A} = \begin{pmatrix} -5 & 0 & -3 & 4 & 2 & -1 \\ 10 & 0 & 6 & -8 & -4 & 2 \end{pmatrix},$$

and thus s_j ($j = 1, 2, \dots, d$) are the columns of the following matrix:

$$S = (3 \ 8 \ 1 \ 4 \ 2 \ 10).$$

We first use the sort algorithm for regularizing $a_1, a_2, \dots, a_6$. Then, we have

$$\begin{aligned} \mathbb{A}_\pi &:= \begin{pmatrix} -5 & -3 & -1 & 0 & 2 & 4 \\ 10 & 6 & 2 & 0 & -4 & -8 \end{pmatrix}, \\ S_\pi &:= (3 \ 1 \ 10 \ 8 \ 2 \ 4), \end{aligned}$$

whose columns $\mathbb{A}_\pi$ and columns S_π are the incidence vectors of the lists $a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(d)}$ and $s_{\pi(1)}, s_{\pi(2)}, \dots, s_{\pi(d)}$, respectively. Now, we apply the forward algorithm for solving SWP with positive multipliers $s_{\pi(1)}, s_{\pi(2)}, \dots, s_{\pi(d)}$. By setting $r = 0$, we obtain

$$\begin{aligned} r &= 0, \quad l = 0, \quad T_0 = 28, \\ k &= 1, \quad T_k = 25, \\ k &= 2, \quad T_k = 21, \\ k &= 3, \quad T_k = 10, \quad l = 1, \quad P(1) = \pi(3), \\ k &= 4, \quad T_k = -8, \quad l = 2, \quad P(2) = \pi(4), \\ k &= 5, \quad T_k = -18. \end{aligned}$$

Hence, the optimal solution set is $[a_{\pi(3)}, a_{\pi(4)}] = [a_6, a_2]$.

Theorem 4.7 *Backward (forward) algorithm works correctly.*

Proof Let $k \in \{1, 2, \dots, d\}$. It is easy to see that the following equalities hold.

$$\begin{aligned} \frac{a_k - a_0}{\|a_k - a_0\|} &= \frac{a_{d+1} - a_0}{\|a_{d+1} - a_0\|}, \\ \frac{a_k - a_{d+1}}{\|a_k - a_{d+1}\|} &= -\frac{a_{d+1} - a_0}{\|a_{d+1} - a_0\|}, \\ \frac{a_k - a_j}{\|a_k - a_j\|} &= \frac{a_k - a_0}{\|a_k - a_0\|} \quad (j = 1, 2, \dots, k-1), \\ \frac{a_k - a_j}{\|a_k - a_j\|} &= \frac{a_k - a_{d+1}}{\|a_k - a_{d+1}\|} \quad (j = k+1, k+2, \dots, d). \end{aligned}$$

If $|T_k| \leq s_k$ ($k = 1, 2, \dots, d$), by the above equalities and Theorem 3.1, the following equivalent statements hold.

$$\begin{aligned} |T_k| \leq s_k &\iff \left\| \sum_{j=k+1}^{d+1} s_j \frac{a_{d+1} - a_0}{\|a_{d+1} - a_0\|} - \sum_{j=0}^{k-1} s_j \frac{a_{d+1} - a_0}{\|a_{d+1} - a_0\|} \right\| \leq s_k \\ &\iff \left\| -\sum_{j=k+1}^{d+1} s_j \frac{a_k - a_{d+1}}{\|a_k - a_{d+1}\|} - \sum_{j=0}^{k-1} s_j \frac{a_k - a_0}{\|a_k - a_0\|} \right\| \leq s_k \\ &\iff \left\| -\sum_{j=k+1}^{d+1} s_j \frac{a_k - a_j}{\|a_k - a_j\|} - \sum_{j=0}^{k-1} s_j \frac{a_k - a_j}{\|a_k - a_j\|} \right\| \leq s_k \\ &\iff -\sum_{\substack{j=1 \\ j \neq k}}^d s_j \frac{a_k - a_j}{\|a_k - a_j\|} \in s_k \partial (\|\cdot\|)(a_k) \end{aligned}$$

$$\begin{aligned} &\Longleftrightarrow -\sum_{\substack{j=1 \\ j \neq k}}^d s_j \frac{a_k - a_j}{\|a_k - a_j\|} \in \partial (s_k \|\cdot\| - a_k) (a_k) \\ &\Longleftrightarrow 0 \in \partial f(a_k). \end{aligned} \quad (4.1)$$

Since the objective function is convex, the proof is complete. $\square$

The proof of Theorem 4.7 leads directly to the following result.

Proposition 4.8 *The optimal solution set of SWP can only be one of the forms a_p or $[a_p, a_{p+1}]$ for some $p \in \{1, 2, \dots, d\}$. In particular, if there exists $p \in \{1, 2, \dots, d\}$ such that the optimal solution of SWP is $[a_p, a_{p+1}]$, then*

$$\begin{aligned} \sum_{j=p+1}^{d+1} s_j - \sum_{j=0}^{p-1} s_j &= s_p, \\ \sum_{j=p+2}^{d+1} s_j - \sum_{j=0}^p s_j &= -s_{p+1}. \end{aligned}$$

Proof Consider the first part of the proof. By contradiction, suppose that there exist $p \in \{1, 2, \dots, d\}$ and $r \in \{2, 3, \dots, d-p\}$ such that $[a_p, a_{p+r}]$ is the optimal solution set of SWP. Therefore, $0 \in \partial f(a_i)$ for $i = p, p+1, \dots, p+r$. Based on (4.1), we have

$$-s_i \leq \sum_{j=i+1}^{d+1} s_j - \sum_{j=0}^{i-1} s_j \leq s_i, \quad i = p, p+1, \dots, p+r.$$

We deduce that

$$\begin{aligned} s_p &\geq \sum_{j=p+1}^{d+1} s_j - \sum_{j=0}^{p-1} s_j \\ &\geq \sum_{\substack{j=p+1 \\ j \neq p+r}}^{d+1} s_j - \sum_{j=0}^{p-1} s_j - \sum_{j=p+r+1}^{d+1} s_j + \sum_{j=0}^{p+r-1} s_j \\ &\geq s_p + 2 \sum_{j=p+1}^{p+r-1} s_j \\ &> s_p. \end{aligned}$$

This is a contradiction and completes the proof.

Now consider the next part of the proof. By contradiction, suppose that

$$\sum_{j=p+1}^{d+1} s_j - \sum_{j=0}^{p-1} s_j \neq s_p.$$

By (4.1), we have

$$\begin{aligned} -s_p &\leq \sum_{j=p+1}^{d+1} s_j - \sum_{j=0}^{p-1} s_j < s_p, \\ -s_{p+1} &\leq \sum_{j=p+2}^{d+1} s_j - \sum_{j=0}^p s_j \leq s_{p+1}. \end{aligned}$$

Therefore,

$$\begin{aligned} s_p &> \sum_{j=p+1}^{d+1} s_j - \sum_{j=0}^{p-1} s_j \\ &\geq \sum_{j=p+2}^{d+1} s_j - \sum_{j=0}^{p-1} s_j - \sum_{j=p+2}^{d+1} s_j + \sum_{j=0}^p s_j \\ &= s_p, \end{aligned}$$

which is a contradiction. Similarly,

$$\sum_{j=p+2}^{d+1} s_j - \sum_{j=0}^p s_j = -s_{p+1}.$$

□

One can apply Proposition 4.8 to determine the optimal solution set of SWP.

Let the list $a_1, a_2, \dots, a_d$ be regular. The point a_i is a median of the list $a_1, a_2, \dots, a_d$ if index i satisfies:

$$\sum_{j=1}^{i-1} s_j < \sum_{j=1}^d \frac{s_j}{2} \quad \text{and} \quad \sum_{j=1}^i s_j \geq \sum_{j=1}^d \frac{s_j}{2}. \quad (4.2)$$

In the collinear case, it is well-known that if a_i be a median of the list $a_1, a_2, \dots, a_d$, then a_i is an optimal solution of SWP (see [2,16,21]).

A necessary and sufficient condition of optimality is presented by Proposition 4.8 for SWP, whereas, Condition 4.2 is only a sufficient condition of optimality for SWP in the collinear case. The following example shows the difference between Condition 4.2 and the presented necessary and sufficient condition of optimality in Proposition 4.8.

Example 4.9 Consider SWP. Let $a_1 = (1, 2)^T$, $a_2 = (2, 4)^T$, $a_3 = (3, 6)^T$, $a_4 = (4, 8)^T$, $s_1 = 1$, $s_2 = 2$, $s_3 = 2$, and $s_4 = 1$. Let us check the optimality condition in the points $x_1 = (2, 4)^T$, $x_2 = (2.25, 4.5)^T$, $x_3 = (3, 6)^T$, $x_4 = (4, 8)^T$.

Consider $x_1 = (2, 4)^T$, then $x_1 = a_2$. In Condition (4.2), we obtain $i = 2$ and

$$\sum_{j=1}^1 s_j = 1 < \sum_{j=1}^4 \frac{s_j}{2} = 3 \quad \text{and} \quad \sum_{j=1}^2 s_j = 3 \geq \sum_{j=1}^4 \frac{s_j}{2} = 3.$$

Therefore, $x_1 = a_2$ satisfies the Condition (4.2), and x_1 is an optimal solution of SWP. Now consider the results in Proposition 4.8, we have

$$T_2 = \sum_{j=3}^5 s_j - \sum_{j=0}^1 s_j = 2.$$

Therefore, $|T_2| \leq 2$ and x_1 is an optimal solution set of SWP.

Now consider $x_2 = (2.25, 4.5)^T$. Since there exists no a_j ($j = 1, 2, \dots, 4$) such that $x_2 = a_j$, Condition (4.2) does not present any information of optimality at this point. Whereas, since $x_2 \in [a_2, a_3]$, from Proposition 4.8, it is necessary and sufficient to show that $T_2 = w_2$ for verifying the optimality condition at x_2 . Since $T_2 = 2$, it follows that x_2 is an optimal solution of SWP.

Now consider $x_3 = (3, 6)^T$. Since $x_3 = a_3$, it is easy to see that $i = 3$. Since

$$\sum_{j=1}^2 s_j = 3 \not\geq \sum_{j=1}^4 \frac{s_j}{2} = 3,$$

it follows that x_3 does not satisfy the Condition (4.2). Whereas $|T_3| = 3 \leq 3$. Therefore, x_3 satisfies the optimality condition in Proposition 4.8. This observation shows that Condition (4.2) is only a sufficient condition of optimality for SWP.

Finally, consider $x_4 = (4, 8)^T$. Since $|T_4| = 5 \not\leq w_4 = 1$, it follows that x_4 is not an optimal solution of SWP. On the other hand,

$$\sum_{j=1}^3 s_j = 5 \not\geq \sum_{j=1}^4 \frac{s_j}{2} = 3.$$

Hence x_4 does not satisfy in Condition (4.2). But Condition (4.2) is only a sufficient condition of optimality, therefore we can not say that x_4 is not an optimal solution of SWP.

Now let us analyze the running time of the backward (forward) algorithm.

Theorem 4.10 *Backward (forward) algorithm terminates after at most d iterations, and can be implemented to run in $O(d)$ time.*

Proof In the worst case, $[a_1, a_2]$ is the optimal solution set and we applied the backward algorithm beginning with $r = d + 1$. Therefore, the backward algorithm terminates with $k = 1$, which implies that the backward algorithm terminates after d iterations. Let us denote the worst case of the running time of backward algorithm for d customers by $g_b(d)$. We obtain

$$g_b(d) = (d + 1) + 2d + 2 = 3d + 3.$$

Similarly, in the worst case of the running time of forward algorithm for d customers (which denotes by $g_f(d)$), the forward algorithm terminates after d iterations and

$$g_f(d) = d + 2d + 2 = 3d + 2.$$

□

We can decrease the overall running time by choosing a suitable initial iteration. For this intention, by Proposition 4.8, we suggest the following approach:

$$\left\{ \begin{array}{l} \text{Use the forward algorithm with } r = \lceil \frac{d}{2} \rceil \text{ if } T_{\lceil \frac{d}{2} \rceil} > s_{\lceil \frac{d}{2} \rceil}. \\ \text{Set } [a_{\lceil \frac{d}{2} \rceil}, a_{\lceil \frac{d}{2} \rceil + 1}] \text{ is the optimal solution set if } T_{\lceil \frac{d}{2} \rceil} = s_{\lceil \frac{d}{2} \rceil}. \\ \text{Set } \{a_{\lceil \frac{d}{2} \rceil}\} \text{ is the optimal solution set if } -s_{\lceil \frac{d}{2} \rceil} < T_{\lceil \frac{d}{2} \rceil} < s_{\lceil \frac{d}{2} \rceil}. \\ \text{Set } [a_{\lceil \frac{d}{2} \rceil - 1}, a_{\lceil \frac{d}{2} \rceil}] \text{ is the optimal solution set if } T_{\lceil \frac{d}{2} \rceil} = -s_{\lceil \frac{d}{2} \rceil}. \\ \text{Use the backward algorithm with } r = \lceil \frac{d}{2} \rceil \text{ if } T_{\lceil \frac{d}{2} \rceil} < -s_{\lceil \frac{d}{2} \rceil}. \end{array} \right.$$

In the worst case of the running time of the above approach, d is odd, $T_{\lceil \frac{d}{2} \rceil} > s_{\lceil \frac{d}{2} \rceil}$, and $\Omega = [a_{d-1}, a_d]$. Let us denote the worst case of the running time of the above approach by $g(d)$, then we obtain

$$g(d) = 2 + d + 2 \left(d - \lceil \frac{d}{2} \rceil \right) + 2 = 2 + d + d + 1 + 2 = 2d + 3.$$

Note that we require $(3n + 2)d - 1$ operations to compute $f(y)$ for any $y \in \mathbb{R}^n$, which is more than the required operations of the forward (backward) algorithm for $n \geq 2$ or $d \geq 2$.

In the next section, we present an example in which the Cooper's algorithm converges to a non-desirable solution. To get over this difficulty, we propose a modified Cooper's algorithm, which improves the Cooper's algorithm towards the desirability of solutions.

5 Modified Cooper's Algorithm

In the following example, we verify the behavior of Cooper's algorithm applied to MWP.

Example 5.1 Assume that there are 12 customers whose locations denoted by a_j ($j = 1, 2, \dots, 12$) are given by the columns of the following matrix:

$$A = \begin{pmatrix} 0 & 2 & 4 & 6 & 8 & 10 & 11 & 13 & 15 & 17 & 19 & 21 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

and all s_j ($j = 1, 2, \dots, 12$) are 1. Suppose that $M = 2$. At the k th iteration, suppose that $A_1^k = \{a_1, a_2, \dots, a_6\}$ and $A_2^k = \{a_7, a_8, \dots, a_{12}\}$. Now, at the $(k+1)$ th iteration of the Cooper's algorithm, the task of location phase is to solve the involved SWPs. In the poor choice, $x_1^{k+1T} = (4, 0)$ and $x_2^{k+1T} = (17, 0)$. Now, if we apply the NCR, we will have $A_i^{k+1} = A_i^k$ ($i = 1, 2$) and the Cooper's algorithm stops with

$$\begin{aligned} F(x_1^{k+1}, x_2^{k+1}) &= \sum_{i=1}^2 \sum_{j=1}^{12} w_{ij} \|x_i^{k+1} - a_j\| \\ &= \sum_{j=1}^6 s_j \|x_1^{k+1} - a_j\| + \sum_{j=7}^{12} s_j \|x_2^{k+1} - a_j\| = 36. \end{aligned}$$

Whereas we can make a correct choice so that the partitions change at the $(k+1)$ th iteration, i.e., new partitions A_i^{k+1} ($i = 1, 2$) are generated by NCR so that there exists $i \in \{1, 2\}$ such that $A_i^{k+1} \neq A_i^k$, and the location phase finds new locations of facilities for these partitions, implying that F decreases strictly. If we set $x_1^{k+1T} = (6, 0)$ and $x_2^{k+1T} = (17, 0)$, then $A_1^{k+1} = \{a_1, a_2, \dots, a_7\}$ and $A_2^{k+1} = \{a_8, a_9, \dots, a_{12}\}$. By applying the location phase, we have $x_1^{k+2T} = (6, 0)$ and $x_2^{k+2T} = (17, 0)$. Hence,

$$\begin{aligned} F(x_1^{k+2}, x_2^{k+2}) &= \sum_{i=1}^2 \sum_{j=1}^{12} w_{ij} \|x_i^{k+2} - a_j\| \\ &= \sum_{j=1}^7 s_j \|x_1^{k+2} - a_j\| + \sum_{j=8}^{12} s_j \|x_2^{k+2} - a_j\| = 35. \end{aligned}$$

We see that $F(x_1^{k+2}, x_2^{k+2}) < F(x_1^{k+1}, x_2^{k+1})$. Indeed, x_1^{k+2} and x_2^{k+2} are more desirable than x_1^{k+1} and x_2^{k+1} .

For simplicity, we denote by Ω_i^k ($i = 1, 2, \dots, M$), the optimal solution set of the involved SWP in the location phase for the cluster A_i^k ($i = 1, 2, \dots, M$), i.e.,

$$\Omega_i^k = \{x^* \in \mathbb{R}^n : F_i(x^*) \leq F_i(x), \forall x \in \mathbb{R}^n\}, i = 1, 2, \dots, M. \quad (5.1)$$

In the location phase in the Cooper's algorithms, the locations of facilities are found by the algorithms in each iteration of the Cooper's algorithms; this may lead to a poor choice of the locations of facilities, which is unavoidable. Therefore, we need

to compute the optimal solution set of the SWPs involved in the location phase. For this intention, if the customers in A_i^k ($i = 1, 2, \dots, M$) are collinear, then we use our scheme for computing Ω_i^k ($i = 1, 2, \dots, M$); otherwise, the objective function of the SWP involved in the location phase for the cluster A_i^k ($i = 1, 2, \dots, M$) is strictly convex function, and Ω_i^k ($i = 1, 2, \dots, M$) is singleton, which can be found by applying one of the known algorithms for solving SWP [1, 12, 14, 18, 19, 27]. Suppose that the Cooper's algorithm is terminated with A_i^k ($i = 1, 2, \dots, M$), then our task is to compute Ω_i^k ($i = 1, 2, \dots, M$) and find $x_i^k \in \Omega_i^k$ ($i = 1, 2, \dots, M$) such that $A_l^{k+1} \neq A_l^k$ for some $l \in \{1, 2, \dots, M\}$, which leads to a decrease in the objective function value.

Moreover, we can use the modified Cooper's algorithm when the Cooper's algorithms are stopped with A_i^k ($i = 1, 2, \dots, M$).

Modified Cooper's algorithm

<i>Input:</i>	The number of customers N , the positive multipliers $s_1, s_2, \dots, s_N$, the location of customers $a_1, a_2, \dots, a_N$, the number of facilities M , the partitions A_i^k ($i = 1, 2, \dots, M$), and the optimal solution sets of involved SWP in the k th iteration Ω_i^k ($i = 1, 2, \dots, M$).
<i>Output:</i>	The desirable solution set of MWP.
<i>Step 1.</i>	Set $t_r := 0, r = 1, 2, \dots, N$.
<i>Step 2.</i>	For $r = 1$ to N set: $\bar{x}_{i,r}^k = \begin{cases} P_{\Omega_i^k}(a_r) & \text{if } a_r \notin A_i^k, \\ \operatorname{argmax}_{\Omega_i^k} \ x - a_r\ & \text{otherwise,} \end{cases} \quad (5.2)$
<i>Step 3.</i>	For $r = 1$ to N do: for $j = 1$ to N do: $d_{i,j}^r = \ \bar{x}_{i,r}^k - a_j\ $ for $i = 1, 2, \dots, M$; if $a_j \in A_h^k$ and $d_{i,j}^r = \min_{i=1,2,\dots,M, i \neq h} \{d_{i,j}^r\} < d_{h,j}^r$, then $A_{h,r}^{k+1} = A_h^k \setminus \{a_j\}$, $A_{h,l}^{k+1} = A_l^k \cup \{a_j\}$, and $t_r = t_r + 1$ (reassign a_j).
<i>Step 4.</i>	If $t_r = 0$ for all $r \in \mathcal{N}$, then $\{\Omega_1^k, \Omega_2^k, \dots, \Omega_M^k\}$ is the desirable solution set of MWP, and the customers in A_i^k ($i = 1, 2, \dots, M$) should be allocated from a $x_i^k \in \Omega_i^k$ ($i = 1, 2, \dots, M$) and stop.
<i>Step 5.</i>	For $r = 1$ to N do: compute $\Omega_{i,r}^{k+1}$ for $i = 1, 2, \dots, M$; where, $\Omega_{i,r}^{k+1}$ is the optimal solution set for the cluster $A_{i,r}^k$, which is defined similar to (5.1).
<i>Step 6.</i>	Let: $\bar{r} = \operatorname{argmin}_{r \in \mathcal{N}} \sum_{i=1}^M \sum_{\{j \in \mathcal{N}, a_j \in A_{i,r}^k\}} s_j \ x_{i,r}^{k+1} - a_j\ ,$ where, $x_{i,r}^{k+1} \in \Omega_{i,r}^{k+1}$ for $i = 1, 2, \dots, M$. Then set $A_i^{k+1} = A_{i,\bar{r}}^{k+1}$, $\Omega_i^{k+1} = \Omega_{i,\bar{r}}^{k+1}$ ($i = 1, 2, \dots, M$), $k := k + 1$, and go to Step 1.

Note that the modified Cooper's algorithm is basically equivalent to the original Cooper's algorithm, if in each iteration the optimal solution sets of involved SWPs

are singletons. In particular, if $t_r = 0$ for some $r \in \mathcal{N}$ in the $(k + 1)$ th iteration, then $\Omega_{i,r}^{k+1} = \Omega_{i,r}^k$. To simplify the modified Cooper's algorithm in Step 2, we present the following proposition, and its proof is easy.

Proposition 5.2 *Let $a, b, c \in \mathbb{R}^n$ and list b, c be regular. Then, the following assertions hold:*

(1) *If $a \notin [b, c]$, then*

$$P_{[b,c]}(a) = \begin{cases} b & \text{if } \bar{t} < 0, \\ b + \bar{t}(c - b) & \text{if } 0 \leq \bar{t} \leq 1, \\ c & \text{if } \bar{t} > 1, \end{cases}$$

where,

$$\bar{t} = \frac{(b - a)^T (b - c)}{\|b - c\|^2}.$$

(2) *Thus,*

$$\max_{x \in [b,c]} \|x - a\| = \max\{\|b - a\|, \|c - a\|\}.$$

One can apply the Proposition 5.2 to solve (5.2) exactly, which decreases sharply the computational time of the Modified Cooper's algorithm.

6 Application

We now proceed to present an application for the importance of finding the desirable solution set of MWP. Consider the following bi-level facility location problem:

$$\min_{y \in \mathbb{R}^n} \sum_{i=1}^M W_i \|y - x_i\|, \quad (6.1)$$

$$(x_1, x_2, \dots, x_M) \in \operatorname{argmin}_{x_i \in \mathbb{R}^n} \left\{ F(x_1, x_2, \dots, x_M) \mid \sum_{i=1}^M w_{ij} = s_j, \forall j \in \mathcal{N} \right\}, \quad (6.2)$$

where,

- (1) y is the location of the wholesale market to be determined;
- (2) W_i is the corresponding weight for the i th facility.

The aim of this model is to find the main source y^* that allocates the demands of the retail markets $x_1, x_2, \dots, x_m$ with the hypothesis that the sum of transportation costs between these retail markets and the demand of the customers be minimized. It is worth mentioning that we need to find a desirable solution of MWP (6.2) for solving the Problem (6.1). Suppose that $\{\Omega_1, \Omega_2, \dots, \Omega_M\}$ is the optimal solution of MWP

Fig. 1 Illustration of Example 6.1

(6.2), then the Problem (6.1) reduces to the following minisum location problem with closest distances (see [5]):

$$\min_{y \in \mathbb{R}^n} \sum_{i=1}^M W_i d_i(y),$$

where, $d_i(y) = \min_{x_i \in \Omega_i} \|x_i - y\|$ for $i = 1, 2, \dots, M$. To make this application clear, the following example is provided.

Example 6.1 Suppose that the locations of customers a_j ($j = 1, 2, \dots, 8$) are given by the columns of the following matrix:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 7 & 7 & 10 & 10 \\ -3 & -2 & 2 & 3 & -3 & 3 & -3 & 3 \end{pmatrix},$$

and all $s_j = 1$. Suppose that $M = 2$ and $W_1 = 2$ and $W_2 = 1$. We first find the optimal solution of the involved MWP in (6.1) by Modified Cooper's algorithm. Therefore, we have $\Omega_1 = \{x \mid x_1 = 0 \text{ and } -2 \leq x_2 \leq 2\}$ and $\Omega_2 = \{(8.5, 0)^T\}$ with $A_1 = \{a_1, a_2, a_3, a_4\}$ and $A_2 = \{a_5, a_6, a_7, a_8\}$. Hence, Problem (6.2) reduces to the following problem:

$$\min_{y \in \mathbb{R}^n} 2d_1(y) + d_2(y). \quad (6.3)$$

It is easy to see that $y^* = (0, 0)^T$ is the optimal solution of Problem (6.3). Moreover, $x_1^* = (0, 0)^T$ and $x_2^* = (8.5, 0)^T$ are the optimal retail markets. Illustrations of solution method for solving Example 6.1 are presented in Fig. 1.

7 Conclusion

In this paper, first we proposed an algorithm for sorting the points in $\mathbb{R}^n$ in the collinear case. Then, using the obtained result, we present an efficient algorithm for solving SWP in the collinear case. Moreover, we modified the Cooper's algorithm which has an advantage in the collinear case. A numerical comparison shows the superiority in efficiency and effectiveness of considering the optimal solution sets of SWPs. The numerical results show that the developed algorithms are suitable for the solution of reasonable MWP with the collinear case. An interesting future research topic is to see whether the results in Sect. 4 can be extended for constrained SWP in the collinear case.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Beck, A., Sabach, S.: Weiszfeld's method: old and new results. *J. Optim. Theory Appl.* **164**(1), 1–40 (2015)
2. Blum, M., Floyd, R.W., Pratt, V.R., Rivest, R.L., Tarjan, R.E.: Time bounds for selection. *J. Comput. Syst. Sci.* **7**(4), 448–461 (1972)
3. Bose, P., Maheshwari, A., Morin, P.: Fast approximations for sums of distances, clustering and the Fermat–Weber problem. *Comput. Geom.* **24**(3), 135–146 (2003)
4. Brimberg, J., Hansen, P., Mladenovic, N., Taillard, E.D.: Improvements and comparison of heuristics for solving the uncapacitated multi-source Weber problem. *Oper. Res.* **48**(3), 444–460 (2000)
5. Brimberg, J., Wesolowsky, G.O.: Locating facilities by minimax relative to closest points of demand areas. *Comput. Oper. Res.* **29**(6), 625–636 (2002)
6. Chandrasekaran, R., Tamir, A.: Open questions concerning Weiszfelds algorithm for the Fermat–Weber location problem. *Math. Program.* **44**(1), 293–295 (1989)
7. Chandrasekaran, R., Tamir, A.: Algebraic optimization: the Fermat–Weber location problem. *Math. Program.* **46**(2), 219–224 (1990)
8. Chatelon, J.A., Hearn, D.W., Lowe, T.J.: A subgradient algorithm for certain minimax and minisum problems. *Math. Program.* **15**(1), 130–145 (1978)
9. Cooper, L.: Heuristic methods for location-allocation problems. *SIAM Rev.* **6**(1), 37–53 (1964)
10. Eyster, J.W., White, J.A., Wierwille, W.W.: On solving multifacility location problems using a hyperboloid approximation procedure. *AIEE Trans.* **5**(1), 1–6 (1973)
11. Gamal, M.D.H., Salhi, S.: A cellular type heuristic for the multisource Weber problem. *Comput. Oper. Res.* **30**(11), 1609–1624 (2003)
12. Görner, S., Kanzow, C.: On Newton's method for the Fermat–Weber location problem. *J. Optim. Theory Appl.* **170**(1), 107–118 (2016)
13. Hansen, P., Mladenovic, N., Taillard, E.: Heuristic solution of the multisource Weber problem as a p-median problem. *Oper. Res. Lett.* **22**(2), 55–62 (1998)

14. Jiang, J.L., Yuan, X.M.: A heuristic algorithm for constrained multi-source Weber problem—the variational inequality approach. *Eur. J. Oper. Res.* **187**(2), 357–370 (2008)
15. Katz, I.N.: Local convergence in Fermat's problem. *Math. Program.* **6**(1), 89–104 (1974)
16. Korte, B., Vygen, J.: *Combinatorial Optimization: Theory and Algorithms*. Springer, Berlin (2002)
17. Kuhn, H.W.: A note on Fermat's problem. *Math. Program.* **4**(1), 98–107 (1973)
18. Levin, Y., Ben-Israel, A.: A heuristic method for large-scale multi-facility location problems. *Comput. Oper. Res.* **31**(2), 257–272 (2004)
19. Li, Y.: A Newton acceleration of the Weiszfeld algorithm for minimizing the sum of Euclidean distances. *Comput. Optim. Appl.* **10**(3), 219–242 (1998)
20. Luis, M., Said, S., Nagy, G.: A guided reactive GRASP for the capacitated multi-source Weber problem. *Comput. Oper. Res.* **38**(7), 1014–1024 (2011)
21. Ostresh, L.M.: On the convergence of a class of iterative methods for solving the Weber location problem. *Oper. Res.* **26**(4), 597–609 (1978)
22. Overton, M.L.: A quadratically convergence method for minimizing a sum of Euclidean norms. *Math. Program.* **27**(1), 34–63 (1983)
23. Rockafellar, R.T.: *Convex Analysis*. Princeton University Press, Princeton (1970)
24. Rosing, K.E.: An optimal method for solving the (generalized) multi-Weber problem. *Eur. J. Oper. Res.* **58**(3), 414–426 (1992)
25. Sherali, H.D., Nordai, F.L.: NP-hard, capacitated, balanced p-median problems on a chain graph with a continuum of link demands. *Math. Oper. Res.* **13**(1), 32–49 (1998)
26. Vardi, Y., Zhang, C.-H.: A modified Weiszfeld algorithm for the Fermat–Weber location problem. *Math. Program.* **90**(3), 559–566 (2001)
27. Weiszfeld, E.: Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Math. J.* **43**, 355–386 (1937)
28. Weiszfeld, E., Plastria, F.: On the point for which the sum of the distances to n given points is minimum. *Ann. Oper. Res.* **167**(1), 7–41 (2009)