Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 18, No. 2, pp. 299–315, 2022

# A SEMISMOOTH NEWTON BASED AUGMENTED LAGRANGIAN ALGORITHM FOR WEBER PROBLEM*

Yong Jin Liu† and Qinxin Zhu

**Abstract:** This paper is concerned with efficient algorithms for solving Weber problem, which is an important problem arising in the facility location problems. In this paper, we reformulate the Weber problem as its equivalent form and then propose a semismooth Newton based augmented Lagrangian (Ssnal) algorithm for solving Weber problem. The global convergence and locally asymptotically superlinear convergence of the Ssnal algorithm are characterized under mild conditions. Numerical experiments conducted on synthetic data sets demonstrate that the Ssnal algorithm outperforms several state-of-the-art algorithms in terms of efficiency and robustness.

**Key words:** *Weber problem, augmented Lagrangian algorithm, semismooth Newton method*

**Mathematics Subject Classification:** *90C06, 90C25, 90C90*

## 1 Introduction

This paper is concerned with the Weber problem, which admits the following form:

$$\min_{x\in\mathbb{R}^n} \quad f(x) = \sum_{i=1}^{m} w_i\|x - a_{[i]}\|, \tag{1.1}$$

where the weights $w_i > 0$, $i = 1, 2, \ldots, m$ are given, the vectors $a_{[1]}$, $a_{[2]}$, ..., $a_{[m]} \in \mathbb{R}^n$ are $m$ mutually distinct points and $||\cdot||$ denotes the $\ell_2$ norm. The Weber problem, one of the best known problems in location theory [7, 26, 31], has been the subject of intense research. Many generalized models based on Weber problem have been proposed, see, for example, [8, 9, 11, 17, 25, 28, 29].

There have been several algorithms for solving Weber problem (1.1). Among them, the Weiszfeld algorithm [39] may be the most popular one as it belongs to the class of fixed-point iteration algorithms, and hence it is easy to be implemented. However, since the objective function of Weber problem is nonsmooth at given points $a_{[i]}$ $(i = 1, \ldots, m)$, a singular situation arises when the iteration points of Weiszfeld algorithm coincide with the given points. Therefore, the global convergence of the Weiszfeld algorithm is not guaranteed. To overcome this deficiency, other algorithms have been proposed. For example, on the basis of

*The work of Yong-Jin Liu was in part supported by the National Natural Science Foundation of China (No. 11871153) and the Natural Science Foundation of Fujian Province of China (No. 2019J01644)

†Corresponding author

Weiszfeld algorithm, Vardi and Zhang [38] propose the modified Weiszfeld (MW) algorithm, which admits global convergence, but increases the computational cost considerably. Görner and Kanzow [10] show that, after a suitable initialization, the standard Newton (NW) algorithm can be applied to the Weber problem. However, when the number of given points is large, the computational cost of NW algorithm is expensive in the initialization. Recently, an alternating direction method of multipliers (ADMM) is proposed in [40] to solve the Weber problem and the numerical results demonstrate the efficiency of ADMM. We refer readers to [12, 13, 14, 19, 22] for more results on Weber problem.

In this paper, we aim to develop an efficient and robust algorithm for solving the Weber problem (1.1). The excellent numerical performance of the semismooth Newton based augmented Lagrangian (Ssnal) algorithm has been demonstrated when it is applied to solve large-scale Lasso problems [20, 21, 24, 41], OSCAR and SLOPE models [27], singly linearly and box constrained least squares regression [23], and support vector machines [32]. The innovation of the Ssnal algorithm is that we can make full use of second-order information when using a semismooth Newton (Ssn) algorithm for its inner subproblem, which improves efficiency of this algorithm. Inspired by success of the Ssnal algorithm, we intend to propose it for solving the Weber problem (1.1).

The main contributions of this paper are summarized as follows. Firstly, we reformulate the Weber problem as an equivalent convex optimization problem (P) and then apply the Ssnal algorithm to solve the problem (P), in which a semismooth Newton algorithm (Ssn) is applied to solve the subproblems. Secondly, the theoretical results on global and local convergence of the Ssnal algorithm are established under mild conditions. Specifically, global convergence of the Ssnal algorithm is guaranteed under the standard stopping criterion for the subproblem, and locally asymptotically superlinear convergence of the Ssnal algorithm is characterized under the quadratic growth condition and the standard stopping criteria for the subproblem. Moreover, the Ssn algorithm is globally convergent and admits fast superlinear or even quadratic convergence rate without any assumption. Finally, we design efficient implementations of the Ssnal algorithm by utilizing special structure of Clarke generalized Jacobian of the relevant proximal mapping. Furthermore, in order to verify robustness and efficiency of the Ssnal algorithm, we report the results of numerical experiments on synthetic data sets by comparing the Ssnal algorithm against state-of-the-art algorithms, including MW, NW, and ADMM.

The rest of this paper is organized as follows. Section 2 presents some preliminaries on the Moreau-Yosida regularization and the subdifferential of $\ell_2$ norm. In Section 3, we develop the Ssnal algorithm to solve the reformulation of the Weber problem (1.1) where the Ssn algorithm is employed to solve its subproblem. Moreover, the theoretical results on the convergence of the Ssnal algorithm are characterized under mild conditions and the theoretical results on the convergence of Ssn algorithm are also given without any assumptions. In Section 4, numerical experiments conducted on synthetic data sets evaluate the performance of our proposed algorithm in comparison with other state-of-the-art algorithms. We make some conclusions in Section 5.

## 2 Preliminaries

In this section, we summarize some notations and present some preliminaries which will be used in the subsequent analysis. Throughout this paper, for given positive integer $n$, we denote $I_n$ as the identity matrix of $n\times n$, where $n$ is often omitted. We define the $\ell_2$ norm unit ball by $\mathcal{B}_2 := \{x \in \mathbb{R}^n \mid \|x\| \leq 1\}$. For any $(\bar{x}, \bar{s}_{[1]}, \ldots, \bar{s}_{[m]}) \in \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n$ and $\varepsilon > 0$, denote $\mathbb{B}_\varepsilon((\bar{x}, \bar{s}_{[1]}, \ldots, \bar{s}_{[m]})) := \{(x, s_{[1]}, \ldots, s_{[m]}) \in \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n \mid \|(x, s_{[1]}, \ldots, s_{[m]}) -$

$(\bar{x}, \bar{s}_{[1]}, \ldots, \bar{s}_{[m]})\| \leq \varepsilon\}$. Moreover, $h^*$ is the Fenchel conjugate of a proper convex function $h$.

Let $h : \mathbb{R}^n \to (-\infty, +\infty]$ be a proper closed convex function. Then, the definitions of proximal mapping and Moreau envelope of $h$ (cf. [30]) are given by respectively

$$\mathrm{Prox}_h(x) := \underset{y \in \mathbb{R}^n}{\mathrm{argmin}}\{h(y) + \frac{1}{2}\|y - x\|^2\},\ \forall x \in \mathbb{R}^n,$$
$$E_h(x) := \min_{y \in \mathbb{R}^n}\{h(y) + \frac{1}{2}\|y - x\|^2\},\ \forall x \in \mathbb{R}^n.$$

For given $t > 0$, it follows from [34] that the following Moreau's identity holds:

$$\mathrm{Prox}_{th}(x) + t\mathrm{Prox}_{h^*/t}(x/t) = x,\ \forall x \in \mathbb{R}^n.$$

It follows from [18] that $E_h(\cdot)$ is convex and continuously differentiable with its gradient

$$\nabla E_h(x) = x - \mathrm{Prox}_h(x),\ \forall x \in \mathbb{R}^n.$$

Furthermore, $\mathrm{Prox}_h(\cdot)$ and $\nabla E_h(\cdot)$ are globally Lipschitz continuous with modulus 1.

For a given closed convex set $\mathcal{Q} \subseteq \mathbb{R}^n$, we denote $\chi_{\mathcal{Q}}$ as the indicator function of $\mathcal{Q}$. If $h = \chi_{\mathcal{Q}}$, the proximal mapping of $h$ at $x$ reduces to the projection of $x$ onto $\mathcal{Q}$, i.e.,

$$\mathrm{Prox}_h(x) = \Pi_{\mathcal{Q}}(x) = \underset{y \in \mathcal{Q}}{\mathrm{argmin}}\{\|y - x\|^2\},\ \forall x \in \mathbb{R}^n.$$

In particular, when $h = \chi_{t\mathcal{B}_2}$ is the indicator function of $t\mathcal{B}_2 = \{y \in \mathbb{R}^n \mid \|y\| \leq t\}$, we have

$$\mathrm{Prox}_h(x) = \Pi_{t\mathcal{B}_2}(x) = \begin{cases} t\dfrac{x}{\|x\|}, & \text{if } \ \|x\| > t, \\ x, & \text{otherwise.} \end{cases} \tag{2.1}$$

The proximal mapping of $\ell_2$ norm is

$$\mathrm{Prox}_{t\|\cdot\|}(x) = \begin{cases} \dfrac{x}{\|x\|}\max\{\|x\| - t, 0\}, & \text{if } \ x \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

**Example 2.1** ([1, Example 3.34])**.** Let $h : \mathbb{R}^n \to \mathbb{R}$ be defined by $h(x) := \|x\|$. Then, the subdifferential set of $h$ at $x \in \mathbb{R}^n$ is given by

$$\partial h(x) = \begin{cases} \{\frac{x}{\|x\|}\}, & x \neq 0, \\ \mathcal{B}_2, & x = 0. \end{cases}$$

Next, we present some definitions on locally Lipschitz function, which will be useful for the rest of our discussions. Let $\mathcal{O} \subseteq \mathbb{R}^{n_1}$ be an open set and $\Psi$: $\mathcal{O} \to \mathbb{R}^{n_2}$ be a locally Lipschitz function. It can be seen from [37] that $\Psi$ is differentiable almost everywhere. Denote $D_\Psi$ as the set of all points where $\Psi$ is differentiable and $J\Psi(x)$ as the Jacobian of $\Psi$ at $x \in D_\Psi$. Then, the B(Bouligand)-subdifferential of $\Psi$ at $x \in \mathbb{R}^{n_1}$ is defined by

$$\partial_B \Psi(x) := \{K \in \mathbb{R}^{n_2 \times n_1} \mid \exists \{x^k\} \subseteq D_\Psi \,\text{such that}\, x^k \to x \text{ and } J\Psi(x^k) \to K\}.$$

Due to [3, Definition 2.6.1], the Clarke generalized Jacobian of $\Psi$ at $x \in \mathbb{R}^{n_1}$ is the convex hull of B-subdifferential of $\Psi$ at $x$, i.e., $\partial\Psi(x) := \mathrm{co}\,(\partial_B\Psi(x))$.

## 3 A Semismooth Newton Based Augmented Lagrangian Algorithm

In this section, we propose a semismooth Newton based augmented Lagrangian (SSNAL) algorithm to solve an equivalent form of the Weber problem (1.1).

We first rewrite Weber problem (1.1) as

$$\begin{aligned} \min_{x, s_{[1]}, \ldots, s_{[m]}} \quad & P(x, s_{[1]}, \ldots, s_{[m]}) := \sum_{i=1}^{m} w_i \|s_{[i]}\| \\ \text{s.t.} \quad & x - a_{[i]} - s_{[i]} = 0,\ i = 1, \ldots, m. \end{aligned} \tag{P}$$

The dual of problem (P) admits the following form:

$$\begin{aligned} \max_{\lambda_{[1]}, \ldots, \lambda_{[m]}} \quad & G(\lambda_{[1]}, \ldots, \lambda_{[m]}) := \sum_{i=1}^{m} \langle a_{[i]}, \lambda_{[i]} \rangle \\ \text{s.t.} \quad & \sum_{i=1}^{m} \lambda_{[i]} = 0, \\ & \|\lambda_{[i]}\| \leq w_i,\ i = 1, \ldots, m. \end{aligned} \tag{D}$$

The Karush-Kuhn-Tucker (KKT) optimality condition of problem (P) is given by

$$\sum_{i=1}^{m} \lambda_{[i]} = 0,\ x - s_{[i]} - a_{[i]} = 0,\ 0 \in \partial(w_i \|s_{[i]}\|) - \lambda_{[i]}, i = 1, 2, \ldots, m.$$

The Lagrangian function of (P) is given by

$$l(x, s_{[1]}, \ldots, s_{[m]}; \lambda_{[1]}, \ldots, \lambda_{[m]}) := \sum_{i=1}^{m} w_i \|s_{[i]}\| + \sum_{i=1}^{m} \langle \lambda_{[i]}, x - s_{[i]} - a_{[i]} \rangle.$$

Furthermore, for given $\sigma > 0$, the augmented Lagrangian function of (P) is

$$\mathcal{L}_\sigma(x, s_{[1]}, \ldots, s_{[m]}, \lambda_{[1]}, \ldots, \lambda_{[m]}) := l(x, s_{[1]}, \ldots, s_{[m]}, \lambda_{[1]}, \ldots, \lambda_{[m]}) + \frac{\sigma}{2} \sum_{i=1}^{m} ||x - s_{[i]} - a_{[i]}||^2.$$

### 3.1 A semismooth Newton based augmented Lagrangian algorithm for problem (P)

In this subsection, we present the framework of a SSNAL algorithm for solving problem (P) and its convergence.

The framework of a SSNAL algorithm for solving problem (P) is outlined below.

**Algorithm 1** (Ssnal) **A semismooth Newton augmented Lagrangian algorithm for** (P)

**Input:** $\sigma_0 > 0$, $(x^0, s^0_{[1]}, \dots, s^0_{[m]}; \lambda^0_{[1]}, \dots, \lambda^0_{[m]}) \in \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n$. Set $k = 0$.

1: Solve approximately

$$x^{k+1} \approx \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \{\varphi_k(x) := \inf_{s_{[1]},\dots,s_{[m]}} \mathcal{L}_{\sigma_k}(x, s_{[1]}, \dots, s_{[m]}; \lambda^k_{[1]}, \dots, \lambda^k_{[m]})\} \tag{3.1}$$

to satisfy the conditions (A) and (B) below.

2: Compute $s^{k+1}_{[i]} = \mathrm{Prox}_{\sigma_k^{-1} w_i \|\cdot\|}(x^{k+1} - a_{[i]} + \sigma_k^{-1} \lambda^k_{[i]})$, $i = 1, \dots, m$.

3: Compute $\lambda^{k+1}_{[i]} = \lambda^k_{[i]} + \sigma_k(x^{k+1} - s^{k+1}_{[i]} - a_{[i]})$, $i = 1, \dots, m$.

4: Update $\sigma_{k+1} \uparrow \sigma_\infty \leq +\infty, k \leftarrow k+1$, and go to Step 1.

Due to [35], we solve approximately (3.1) under the following stopping criteria:

$$\varphi_k(x^{k+1}) - \inf \varphi_k \leq \frac{\varepsilon_k^2}{2\sigma_k}, \quad \sum_{k=0}^{\infty} \varepsilon_k < \infty, \tag{A}$$

$$\varphi_k(x^{k+1}) - \inf \varphi_k \leq \frac{\delta_k^2}{2\sigma_k} \|(\lambda^{k+1}_{[1]}, \dots, \lambda^{k+1}_{[m]}) - (\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\|^2, \quad \sum_{k=0}^{\infty} \delta_k < \infty, \tag{B}$$

$$\|\nabla \varphi_k(x^{k+1})\| \leq \frac{\delta'_k}{\sigma_k} \|(\lambda^{k+1}_{[1]}, \dots, \lambda^{k+1}_{[m]}) - (\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\|, \quad 0 \leq \delta'_k \to 0, \tag{B'}$$

where $\{\varepsilon_k\}$, $\{\delta_k\}$ and $\{\delta'_k\}$ are given nonnegative error tolerance sequences.

Next, we wish to give the results on global and local convergence of the Ssnal algorithm. Since the optimal solution set of problem (P) is nonempty, the global convergence of Algorithm 1 can be obtained directly from [35, 36].

**Theorem 3.1** (Global convergence)**.** *Let* $\{(x^k, s^k_{[1]}, \dots, s^k_{[m]}, \lambda^k_{[1]}, \dots, \lambda^k_{[m]})\}$ *be the infinite sequence generated by Algorithm 1 with stopping criterion* (A)*. Then, the sequence* $\{(x^k, s^k_{[1]}, \dots, s^k_{[m]})\}$ *converges to an optimal solution of* (P) *and the sequence* $\{(\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\}$ *converges to an optimal solution of* (D)*.*

Here, we present the results related to local convergence of the Ssnal algorithm. For this purpose, we give some notions and definitions. The essential objective functions of (P) and (D) are given by

$$\begin{aligned} p(x, s_{[1]}, \dots, s_{[m]}) &:= \sup_{\lambda_{[1]},\cdots,\lambda_{[m]} \in \mathbb{R}^n} l(x, s_{[1]}, \dots, s_{[m]}; \lambda_{[1]}, \dots, \lambda_{[m]}) \\ &= \begin{cases} P(x, s_{[1]}, \dots, s_{[m]}), & x - s_{[i]} - a_{[i]} = 0, i = 1, \dots, m, \\ +\infty, & \text{otherwise}, \end{cases} \\ g(\lambda_{[1]}, \dots, \lambda_{[m]}) &:= \inf_{x, s_{[1]},\cdots,s_{[m]} \in \mathbb{R}^n} l(x, s_{[1]}, \dots, s_{[m]}; \lambda_{[1]}, \dots, \lambda_{[m]}) \\ &= \begin{cases} G(\lambda_{[1]}, \dots, \lambda_{[m]}), & \sum_{i=1}^m \lambda_{[i]} = 0, \|\lambda_{[i]}\| \leq w_i, i = 1, \dots, m, \\ -\infty, & \text{otherwise}. \end{cases} \end{aligned}$$

Define the following maximal monotone operators [34, 35]:

$$\begin{aligned}&\mathcal{T}_p(x,s_{[1]},\dots,s_{[m]}):=\partial p(x,s_{[1]},\dots,s_{[m]}),\quad \mathcal{T}_g(\lambda_{[1]},\dots,\lambda_{[m]}):=\partial g(\lambda_{[1]},\dots,\lambda_{[m]}),\\ &\mathcal{T}_l(x,s_{[1]},\dots,s_{[m]};\lambda_{[1]},\dots,\lambda_{[m]}):=\{(x',s'_{[1]},\dots,s'_{[m]};\lambda'_{[1]},\dots,\lambda'_{[m]})\mid\\ &\qquad (x',s'_{[1]},\dots,s'_{[m]};-\lambda'_{[1]},\dots,-\lambda'_{[m]})\in\partial l(x,s_{[1]},\dots,s_{[m]};\lambda_{[1]},\dots,\lambda_{[m]})\}.\end{aligned}$$

Denote $\mathcal{T}_p^{-1}$, $\mathcal{T}_g^{-1}$, $\mathcal{T}_l^{-1}$ as the inverse of $\mathcal{T}_p$, $\mathcal{T}_g$, $\mathcal{T}_l$ respectively, i.e.,

$$\begin{aligned}&\mathcal{T}_p^{-1}(x,s_{[1]},\dots,s_{[m]}):=\partial p^*(x,s_{[1]},\dots,s_{[m]}),\quad \mathcal{T}_g^{-1}(\lambda_{[1]},\dots,\lambda_{[m]}):=\partial g^*(\lambda_{[1]},\dots,\lambda_{[m]}),\\ &\mathcal{T}_l^{-1}(x',s'_{[1]},\dots,s'_{[m]};\lambda'_{[1]},\dots,\lambda'_{[m]}):=\{(x,s_{[1]},\dots,s_{[m]};-\lambda_{[1]},\dots,-\lambda_{[m]})\mid\\ &\qquad (x',s'_{[1]},\dots,s'_{[m]};\lambda'_{[1]},\dots,\lambda'_{[m]})\in\partial l(x,s_{[1]},\dots,s_{[m]};\lambda_{[1]},\dots,\lambda_{[m]})\}.\end{aligned}$$

Denote $\Omega_P$ and $\Omega_D$ as the set of optimal solutions for (P) and (D), respectively. Let $\mathrm{F_P}$ and $\mathrm{F_D}$ be the set of feasible points for (P) and (D), respectively, i.e.,

$$\begin{aligned}&\mathrm{F_P}:=\{(x,s_{[1]},\dots,s_{[m]})\in\mathbb{R}^n\times\mathbb{R}^n\times\cdots\times\mathbb{R}^n\mid x-s_{[i]}-a_{[i]}=0,i=1,\dots,m\},\\ &\mathrm{F_D}:=\{(\lambda_{[1]},\dots,\lambda_{[m]})\in\mathbb{R}^n\times\cdots\times\mathbb{R}^n\mid\sum_{i=1}^m\lambda_{[i]}=0;\ \|\lambda_{[i]}\|\le w_i,i=1,\dots,m\}.\end{aligned}$$

It is said that the quadratic growth condition [4, 5] of (P) holds at $(\bar{x},\bar{s}_{[1]},\dots,\bar{s}_{[m]})\in\Omega_P$, if there exist constants $\kappa_p$ and $\varepsilon_p>0$ such that

$$\begin{aligned}&P(x,s_{[1]},\dots,s_{[m]})\ge P(\bar{x},\bar{s}_{[1]},\dots,\bar{s}_{[m]})+\kappa_p\mathrm{dist}^2((x,s_{[1]},\dots,s_{[m]}),\Omega_P),\\ &\qquad\qquad\forall(x,s_{[1]},\dots,s_{[m]})\in\mathrm{F_P}\cap\mathbb{B}_{\varepsilon_p}((\bar{x},\bar{s}_{[1]},\dots,\bar{s}_{[m]})).\end{aligned}$$

The quadratic growth condition for (D) at $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})\in\Omega_D$ is said to hold if there exist positive constants $\kappa_d$ and $\varepsilon_d$ such that

$$\begin{aligned}&-G(\lambda_{[1]},\dots,\lambda_{[m]})\ge -G(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})+\kappa_d\mathrm{dist}^2((\lambda_{[1]},\dots,\lambda_{[m]}),\Omega_D),\\ &\qquad\qquad\forall(\lambda_{[1]},\dots,\lambda_{[m]})\in\mathrm{F_D}\cap\mathbb{B}_{\varepsilon_d}((\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})).\end{aligned}\tag{3.2}$$

The constants $\kappa_p$ and $\kappa_d$ are called the quadratic growth modulus for (P) at $(\bar{x},\bar{s}_{[1]},\dots,\bar{s}_{[m]})$ and for (D) at $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})$, respectively.

A mapping $\Gamma$: $\mathbb{R}^{n_1}\rightrightarrows\mathbb{R}^{n_2}$ is said to be calm [5, 6] with modulus $\kappa$ at $\bar{u}\in\mathbb{R}^{n_1}$ for $\bar{v}\in\mathbb{R}^{n_2}$ if $(\bar{u},\bar{v})\in\mathrm{gph}\,\Gamma$ and there exist positive constants $\varepsilon$ and $\delta$ such that

$$\Gamma(u)\cap\mathbb{B}_\delta(\bar{v})\subseteq\Gamma(\bar{u})+\kappa\|u-\bar{u}\|\mathcal{B}_2,\ \forall u\in\mathbb{B}_\varepsilon(\bar{u}),$$

where $\mathrm{gph}\,\Gamma$ denotes the graph of $\Gamma$.

By virtue of [4, 5, 6], we obtain the following results.

**Proposition 3.2.** *$T_g^{-1}$ is calm at the origin for $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})$ if and only if the quadratic growth condition* (3.2) *for* (D) *holds at $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})$. Specifically, if* (3.2) *holds with quadratic growth modulus $\kappa$, then $T_g^{-1}$ is calm at the origin for $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})$ with modulus $1/\kappa$. Conversely, if $T_g^{-1}$ is calm at the origin for $(\bar{\lambda}_{[1]},\dots,\bar{\lambda}_{[m]})$ with modulus $\kappa'$, then* (3.2) *holds for any $\kappa\in(0,1/(4\kappa'))$.*

Combining Proposition 3.2 with [5], we are able to state local convergence of the Ssnal algorithm.

**Theorem 3.3** (Local convergence)**.** *Let* $\{(x^k, s^k_{[1]}, \dots, s^k_{[m]}, \lambda^k_{[1]}, \dots, \lambda^k_{[m]})\}$ *be an infinite sequence generated by the* SSNAL *algorithm for* (P) *under criterion* (A) *and* $\{(\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\}$ *converge to* $\{(\lambda^\infty_{[1]}, \dots, \lambda^\infty_{[m]})\}$*. If criterion* (B) *is also executed in the* SSNAL *algorithm and the quadratic growth condition* (3.2) *holds at* $\{(\lambda^\infty_{[1]}, \dots, \lambda^\infty_{[m]})\}$ *with modulus* $\kappa$*. Then, for sufficiently large* $k$*,*

$$\mathrm{dist}((\lambda^{k+1}_{[1]}, \dots, \lambda^{k+1}_{[m]}), \mathcal{T}_g^{-1}(0)) \le \mu_k \mathrm{dist}((\lambda^k_{[1]}, \dots, \lambda^k_{[m]}), \mathcal{T}_g^{-1}(0)),$$

*where* $\mu_k = [\delta_k + (\delta_k + 1)\kappa/\sqrt{\kappa^2 + \sigma_k^2}]/(1 - \delta_k) \to \mu_\infty := \kappa/\sqrt{\kappa^2 + \sigma_\infty^2}$*.*

*If in addition to* (A), (B)*, and* (3.2) *holds at* $\{(\lambda^\infty_{[1]}, \dots, \lambda^\infty_{[m]})\}$*, one has criterion* (B') *and* $\mathcal{T}_l^{-1}$ *is upper Lipschitz continuous at the origin with modulus* $\kappa_l$*. Then, for sufficiently large* $k$*,*

$$\mathrm{dist}((x^{k+1}, s^{k+1}_{[1]}, \dots, s^{k+1}_{[m]}), \mathcal{T}_p^{-1}(0)) \le \mu'_k \|(\lambda^{k+1}_{[1]}, \dots, \lambda^{k+1}_{[m]}) - (\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\|,$$

*where* $\mu'_k = (\kappa_l/\sigma_k)(1 + \delta'^2_k) \to \mu'_\infty := \kappa_l/\sigma_\infty$*.*

*Proof.* From Proposition 3.2, we know that $T_g^{-1}$ is calm at the origin for $\{(\lambda^\infty_{[1]}, \dots, \lambda^\infty_{[m]})\}$ with modulus $1/\kappa$, if the quadratic growth condition (3.2) holds at $\{(\lambda^\infty_{[1]}, \dots, \lambda^\infty_{[m]})\}$ with modulus $\kappa$. Therefore, the first part follows from [5, Proposition 3(a)] and the other part of the proof comes from [5, Proposition 3(b)]. □

## 3.2 A semismooth Newton algorithm for the subproblem

In this subsection, we apply an efficient semismooth Newton algorithm [15, 16, 33] to solve the subproblem (3.1).

For given $\sigma > 0$ and $(\tilde{\lambda}_{[1]}, \dots, \tilde{\lambda}_{[m]}) \in \mathbb{R}^n \times \cdots \times \mathbb{R}^n$, we consider the following minimization problem

$$\min_{x \in \mathbb{R}^n} \{\varphi(x) := \inf_{s_{[1]}, \dots, s_{[m]}} \mathcal{L}_\sigma(x, s_{[1]}, \dots, s_{[m]}; \tilde{\lambda}_{[1]}, \dots, \tilde{\lambda}_{[m]})\},$$

where $\varphi(\cdot)$ has the following expression:

$$\begin{aligned}\varphi(x) &= \inf_{s_{[1]}, \dots, s_{[m]}} \mathcal{L}_\sigma(x, s_{[1]}, \dots, s_{[m]}, \tilde{\lambda}_{[1]}, \dots, \tilde{\lambda}_{[m]}) \\ &= \inf_{s_{[1]}, \dots, s_{[m]}} \left\{ \sum_{i=1}^m \left[ w_i \|s_{[i]}\| + \frac{\sigma}{2} ||s_{[i]} - (x - a_{[i]} + \sigma^{-1}\tilde{\lambda}_{[i]})||^2 \right] \right\} - \frac{1}{2\sigma} \sum_{i=1}^m ||\tilde{\lambda}_{[i]}||^2 \\ &= \sum_{i=1}^m \sigma E_{\sigma^{-1} w_i \|\cdot\|}(x - a_{[i]} + \sigma^{-1}\tilde{\lambda}_{[i]}) - \frac{1}{2\sigma} \sum_{i=1}^m ||\sigma^{-1}\tilde{\lambda}_{[i]}||^2.\end{aligned}$$

One obtains that $\varphi(\cdot)$ is convex and continuously differentiable due to the fact that Moreau envelope $E_{\sigma^{-1} w_i \|\cdot\|}(\cdot)$ is convex and continuously differentiable. Thus, the solution of subproblem (3.1) can be obtained by solving the following nonsmooth equations:

$$\begin{aligned}0 = \nabla\varphi(x) &= \sum_{i=1}^m \sigma(x - a_{[i]} + \sigma^{-1}\tilde{\lambda}_{[i]} - \mathrm{Prox}_{\sigma^{-1} w_i ||\cdot||}(x - a_{[i]} + \sigma^{-1}\tilde{\lambda}_{[i]})) \\ &= \sum_{i=1}^m \mathrm{Prox}_{\sigma(w_i||\cdot||)^*}(\sigma x - \sigma a_{[i]} + \tilde{\lambda}_{[i]}) \\ &= \sum_{i=1}^m \Pi_{w_i \mathcal{B}_2}(\sigma x - \sigma a_{[i]} + \tilde{\lambda}_{[i]}).\end{aligned}$$

Since $\Pi_{w_i\|\cdot\|}(\cdot)$ is Lipschitz continuous, we have

$$\hat{\partial}^2\varphi(x) := \sigma\partial\left[\sum_{i=1}^{m}\Pi_{w_i\mathcal{B}_2}(\sigma x - \sigma a_{[i]} + \tilde{\lambda}_{[i]})\right].$$

Denote $z_{[i]} = \sigma x - \sigma a_{[i]} + \tilde{\lambda}_{[i]}$, $i = 1, \ldots, m$. Then, from (2.1), the projection onto $\ell_2$ norm ball and its Clarke generalized Jacobian are given by

$$\Pi_{w_i\mathcal{B}_2}(z_{[i]}) = \begin{cases} w_i\dfrac{z_{[i]}}{\|z_{[i]}\|}, & \|z_{[i]}\| > w_i, \\ z_{[i]}, & \text{otherwise}, \end{cases}$$

$$\partial\Pi_{w_i\mathcal{B}_2}(z_{[i]}) = \begin{cases} \left\{\dfrac{w_i}{\|z_{[i]}\|}\left(I - \dfrac{(z_{[i]})(z_{[i]})^T}{\|z_{[i]}\|^2}\right)\right\}, & \|z_{[i]}\| > w_i, \\ \left\{I - t\dfrac{(z_{[i]})(z_{[i]})^T}{(w_i)^2} \,|\, 0 \le t \le 1\right\}, & \|z_{[i]}\| = w_i, \\ \{I\}, & \|z_{[i]}\| < w_i. \end{cases}$$

Let $W \in \partial\left(\sum_{i=1}^{m}\Pi_{w_i\mathcal{B}_2}(z_{[i]})\right)$, one easily obtains that

$$W = \sum_{i\in\mathcal{I}_1}\left[\frac{w_i}{\|z_{[i]}\|}\left(I - \frac{(z_{[i]})(z_{[i]})^T}{\|z_{[i]}\|^2}\right)\right] + \sum_{i\in\mathcal{I}_2}\left[I - t_i\frac{(z_{[i]})(z_{[i]})^T}{(w_i)^2}\right] + \sum_{i\in\mathcal{I}_3} I, \tag{3.3}$$

where $\mathcal{I}_1$, $\mathcal{I}_2$ and $\mathcal{I}_3$ are the index sets defined by

$$\mathcal{I}_1 := \{i \,|\, \|z_{[i]}\| > w_i\},\ \mathcal{I}_2 := \{i \,|\, \|z_{[i]}\| = w_i\},\ \mathcal{I}_3 := \{i \,|\, \|z_{[i]}\| < w_i\}$$

and $t_i \in [0,1], i \in \mathcal{I}_2$. It is obvious that

$$V := \sigma W \in \hat{\partial}^2\varphi(x).$$

From [3, Proposition 2.3.3 and Theorem 2.6.6], we know that $\partial^2\varphi(x) \subseteq \hat{\partial}^2\varphi(x)$, where $\partial^2\varphi(x)$ is the generalized Hessian of $\varphi$ at $x$.

Since $\Pi_{\mathcal{B}_2}$ is strongly semismooth, we know that $\nabla\varphi(\cdot)$ is strongly semismooth. Therefore, we apply a semismooth Newton algorithm to solve the nonsmooth equations $\nabla\varphi(x) = 0$, which is expected to attain a fast superlinear or even quadratic rate of convergence.

**Algorithm 2** (SSN) **A semismooth Newton algorithm for problem** (3.1)

**Input:** $\mu \in (0, 1/2)$, $\eta \in (0, 1]$, $\gamma_1, \gamma_2 \in (0, 1)$, $\tau \in (0, 1]$, $\rho \in (0, 1)$, $x^0 \in \mathbb{R}^n$. Set $j = 0$.

1: Choose $V_j \in \hat{\partial}^2 \varphi(x^j)$. Apply the conjugate gradient (CG) algorithm to solve the linear system

$$(V_j + \epsilon_j I)d = -\nabla \varphi(x^j),$$

to find $d^j$ such that

$$\|(V_j + \epsilon_j I)d^j + \nabla \varphi(x^j)\| \leq \min\{\eta, \|\nabla \varphi(x^j)\|^{1+\tau}\},$$

where $\epsilon_j := \min\{\gamma_1, \gamma_2 \|\nabla \varphi(x^j)\|\}$.

2: Set $\alpha_j = \rho^{m_j}$, where $m_j$ is the smallest nonnegative integer $m$ satisfying

$$\varphi(x^j + \rho^m d^j) \leq \varphi(x^j) + \mu \rho^m \langle \nabla \varphi(x^j), d^j \rangle.$$

3: Set $x^{j+1} = x^j + \alpha_j d^j$, $j = j + 1$, and go to Step 1.

Next, according to [42, Theorem 3.4 and 3.5], we present the results on global convergence and superlinear convergence of Algorithm 2.

**Theorem 3.4.** *Let $\{x^j\}$ be the infinite sequence generated by Algorithm 2. Then, $\{x^j\}$ converges to an optimal solution $\hat{x}$ of problem* (3.1). *Furthermore, the rate of convergence is at least superlinear with*

$$\|x^{j+1} - x^j\| = O(\|x^j - \hat{x}\|^{1+\tau}),$$

*where $\tau$ is the parameter used in Algorithm 2.*

## 4 Numerical Experiments

In this section, we carry out numerical experiments in order to evaluate the performance of the SSNAL algorithm for solving the Weber problem (1.1) on synthetic data sets. We compare the SSNAL algorithm with several state-of-the-art algorithms, including the modified Weiszfeld (MW) algorithm [38], Newton (NW) algorithm [10] and alternating direction method of multipliers (ADMM) [40]. All our experiments are executed in MATLAB R2019a on a Windows workstation with Intel Xeon Gold 6144 CPU at 3.50 GHz and 256 GB memory.

### 4.1 Some existing algorithms for solving Weber problem

In this subsection, we briefly present some algorithmic frameworks including MW, NW, and ADMM.

#### 4.1.1 Weiszfeld algorithm and modified Weiszfeld algorithm

According to [2, 39], the Weiszfeld algorithm is iterated as follows:

$$x^{k+1} = \frac{\sum_{i=1}^m w_i \|x^k - a_{[i]}\|^{-1} a_{[i]}}{\sum_{i=1}^m w_i \|x^k - a_{[i]}\|^{-1}}. \tag{4.1}$$

From (4.1), one obtains that when the iteration point $x^k$ coincides with $a_{[i]}$, the algorithm would be forced to terminate and hence its convergence cannot be guaranteed. To overcome the drawback, a modified Weiszfeld (MW) algorithm is proposed in [38] and its framework is given below.

---

**Algorithm 3 (MW) A modified Weiszfeld algorithm for problem** (1.1)

---

**Input:** $x^0 \in \mathbb{R}^n$. Set $k$=0.

1: Compute

$$x^{k+1} = \left(1 - \frac{\eta(x^k)}{r(x^k)}\right)^+ \tilde{T}(x^k) + \min\left\{1, \frac{\eta(x^k)}{r(x^k)}\right\} x^k.$$

2: Set $k = k + 1$, and go to Step 1.

---

In Algorithm 3, $\tilde{T}(x)$, $\eta(x)$ and $r(x)$ are defined by

$$\tilde{T}(x) = \frac{\sum\limits_{i:a_{[i]}\neq x} w_i \|x^k - a_{[i]}\|^{-1} a_{[i]}}{\sum\limits_{i:a_{[i]}\neq x} w_i \|x^k - a_{[i]}\|^{-1}}, \quad \eta(x) = \begin{cases} w_i & \text{if } \ x = a_{[i]}, i = 1, 2, \ldots, m, \\ 0 & \text{otherwise,} \end{cases}$$

$$R(x) = \sum_{i:a_{[i]}\neq x} w_i \frac{x - a_{[i]}}{\|x - a_{[i]}\|}, \quad r(x) = \|R(x)\|.$$

From Algorithm 3, we know that the modified Weiszfeld algorithm produces the same iteration point as the Weiszfeld algorithm when $x^k \notin \{a_{[1]}, a_{[2]}, \ldots, a_{[m]}\}$.

### 4.1.2 Newton algorithm

The algorithmic framework of Newton (NW) algorithm for solving Weber problem (1.1) is outlined in Algorithm 4.

---

**Algorithm 4 (NW) Newton algorithm for problem** (1.1)

---

**Initialize:** Input $\rho \in (0, 1)$, $\sigma \in (0, 1/2)$. Set $k = 0$. Determine $p \in \{1, 2, \ldots, m\}$ such that $f(a_p) = \min\{f(a_1), \ldots, f(a_m)\}$. If $a_p$ satisfies

$$\left\| \sum_{i=1;i\neq p}^{m} w_i \frac{a_{[p]} - a_{[i]}}{\|a_{[p]} - a_{[i]}\|} \right\| \leq w_p,$$

the algorithm terminates. Otherwise, set $x^0 = a_p + t_p d^p$, and go to Step 1.

1: Compute $d^k$ by solving

$$\nabla^2 f(x^k) d^k = -\nabla f(x^k).$$

2: Compute $t_k$ as the largest number in $\{1, \rho, \rho^2, \ldots\}$ such that

$$f(x^k + t_k d^k) \leq f(x^k) + \sigma t_k \nabla f(x^k)^T d^k.$$

3: Set $x^{k+1} = x^k + t_k d^k$, $k \leftarrow k + 1$, and go to Step 1.

---

In Algorithm 4, $d^p$ and $t_p$ are defined by

$$d^p := -\frac{R_{[p]}}{\|R_{[p]}\|} \quad \text{with} \quad R_{[p]} = \sum_{i=1;i\neq p}^{m} w_i \frac{a_{[p]} - a_{[i]}}{\|a_{[p]} - a_{[i]}\|},$$
$$t_p := \frac{(\|R_{[p]}\| - w_p)}{L(a_{[p]})} \quad \text{with} \quad L(a_{[p]}) = \sum_{i=1;i\neq p}^{m} \frac{w_i}{\|a_{[p]} - a_{[i]}\|}.$$

### 4.1.3 Alternating direction method of multipliers

The alternating direction method of multipliers (ADMM) is applied to solve problem (P), which is stated in Algorithm 5.

**Algorithm 5 (ADMM) Alternating direction method of multipliers for problem** (P)

**Input:** $\sigma_0 > 0$, $(x^0, s^0_{[1]}, \dots, s^0_{[m]}; \lambda^0_{[1]}, \dots, \lambda^0_{[m]}) \in \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n$. Set $k = 0$.

1: Compute $x^{k+1} = \frac{1}{m}\sum_{i=1}^{m}(a_{[i]} + s^k_{[i]} - \frac{1}{\sigma}\lambda^k_{[i]})$.
2: Compute $s^{k+1}_{[i]} = \text{Prox}_{\sigma^{-1}w_i\|\cdot\|}(x^{k+1} - a_{[i]} + \frac{1}{\sigma}\lambda^k_{[i]}), i = 1, 2, \dots, m$.
3: Compute $\lambda^{k+1}_{[i]} = \lambda^k_{[i]} + \sigma(x^{k+1} - a_{[i]} - s^{k+1}_{[i]}), i = 1, 2, \dots, m$.
4: Set $k \leftarrow k + 1$, and go to Step 1.

### 4.2 Stopping criteria

In our numerical experiments, based on the KKT conditions of problem (P), we measure accuracy of the approximate solution obtained by the Ssnal algorithm using the following relative residuals:

$$\begin{aligned}
\text{Res}_1 &:= \frac{\left\|\sum_{i=1}^{m} \lambda_{[i]}\right\|}{1 + \|\lambda_{[1]}\| + \cdots + \|\lambda_{[m]}\|},\\
\text{Res}_2 &:= \max\left\{\|\lambda_{[1]}\| - w_1, \dots, \|\lambda_{[m]}\| - w_m, 0\right\},\\
\text{Res}_3 &:= \max\left\{\frac{\left\|x - a_{[1]} - s_{[1]}\right\|}{1 + \left\|a_{[1]}\right\| + \left\|s_{[1]}\right\|}, \dots, \frac{\left\|x - a_{[m]} - s_{[m]}\right\|}{1 + \left\|a_{[m]}\right\| + \left\|s_{[m]}\right\|}\right\}.
\end{aligned}$$

We choose $(x, s_{[1]}, \dots, s_{[m]};\ \lambda_{[1]}, \dots, \lambda_{[m]}) = (0, 0, \dots, 0; 0, \dots, 0)$ as the initial point of the Ssnal algorithm and terminate the algorithm when

$$\text{Res} := \max\{\text{Res}_1, \text{Res}_2, \text{Res}_3\} \leq \text{tol}, \tag{4.2}$$

where "tol" is a given tolerance.

Similarly, we initialize the ADMM algorithm with $(x, s_{[1]}, \dots, s_{[m]};\ \lambda_{[1]}, \dots, \lambda_{[m]}) = (0, 0, \dots, 0; 0, \dots, 0)$ and terminate the algorithm when

$$\text{Res} := \max\{\text{Res}_1, \text{Res}_2, \text{Res}_3\} \leq \text{tol}.$$

By contrast, since the MW algorithm do not produce the dual sequence $\{(\lambda^k_{[1]}, \dots, \lambda^k_{[m]})\}$, the above stopping criterion is not suitable. Because the MW algorithm may generate singular points during the computation, we terminate the MW algorithm with the subdifferential of the objective function of the Weber problem (1.1). The subdifferential of (1.1) is

$$\partial f(x) = \left\{ \sum_{j\in\mathcal{J}_1} w_j \frac{x - a_{[j]}}{\|x - a_{[j]}\|} + \sum_{j\in\mathcal{J}_2} w_j \xi_j \,|\, \xi_j \in \mathcal{B}_2 \right\},$$

where $\mathcal{J}_1$ and $\mathcal{J}_2$ are defined by

$$\mathcal{J}_1 = \{j \,|\, x \neq a_{[j]}\},\ \mathcal{J}_2 = \{j \,|\, x = a_{[j]}\},\ j = 1,\dots,m.$$

If $\mathcal{J}_2$ is empty, $\partial f(x) = \nabla f(x)$. Otherwise, there exists a unique element in $\mathcal{J}_2$. Without loss of generality, let $\mathcal{J}_2 = \{q\}$. By the optimality condition, we have

$$x - \mathrm{Prox}_{w_q\|\cdot\|}\left(x - \sum_{j\in\mathcal{J}_1} w_j \frac{x - a_{[j]}}{\|x - a_{[j]}\|}\right) = 0.$$

Then, one obtains that

$$\mathrm{res} = \begin{cases} \sum_{j=1}^m w_j \dfrac{x - a_{[j]}}{\|x - a_{[j]}\|}, & \mathcal{J}_2 = \emptyset, \\ x - \mathrm{Prox}_{w_q\|\cdot\|}\left(x - \sum_{j\in\mathcal{J}_1} w_j \dfrac{x - a_{[j]}}{\|x - a_{[j]}\|}\right), & \text{otherwise.} \end{cases}$$

Therefore, we initialize the MW algorithm with $x = 0$ and terminate it when

$$\mathrm{Res} := \|\mathrm{res}\| \leq \mathrm{tol}.$$

Likewise, the stopping criterion (4.2) can not be applied to the NW algorithm since it is also unable to produce the dual sequence $\{(\lambda_{[1]}^k, \dots, \lambda_{[m]}^k)\}$. Since the NW algorithm does not generate singular points during the computation, we terminate the NW algorithm using the following stopping criterion

$$\mathrm{Res} := \|\nabla f(x)\| \leq \mathrm{tol},$$

where $\nabla f(x)$ is given by

$$\nabla f(x) = \sum_{j=1}^m w_j \frac{x - a_{[j]}}{\|x - a_{[j]}\|}.$$

In addition, all tested algorithms terminate when they reach the preset maximum number of iterations (100 for SSNAL and 20000 for MW, NW and ADMM) or the maximum running time of 3 hours.

## 4.3 Numerical results

In this section, we compare the SSNAL algorithm with the MW, NW, and ADMM algorithms for solving Weber problem.

For SSNAL in Algorithm 1, we set the penalty parameter $\sigma_0 = \min\{m/\sqrt{n}, 800\}$. For SSN in Algorithm 2, we choose $t_i = 0$ in (3.3) when $\|z_{[i]}\| = w_i$ $(i = 1,\dots,m)$. Then,

$$W = \sum_{i\in\mathcal{C}_1} \left[ \frac{w_i}{\|z_{[i]}\|} \left( I - \frac{(z_{[i]})(z_{[i]})^T}{\|z_{[i]}\|^2} \right) \right] + \sum_{i\in\mathcal{C}_2} I,$$

where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the index sets defined by

$$\mathcal{C}_1 = \{i \,|\, \|z_{[i]}\| > w_i\},\ \mathcal{C}_2 = \{i \,|\, \|z_{[i]}\| \leq w_i\}.$$

Meanwhile, we set $\mu = 10^{-12}$ and $\rho = 0.5$. The stopping criterion of CG algorithm at the $j$-th iteration of SSN algorithm is chosen as $\|(V_j + \epsilon_j I)d^j + \nabla\varphi(x^j)\| \le {\text{tol}_{\text{cg}}}^j$ with

$$
{\text{tol}_{\text{cg}}}^j = \begin{cases} 0.01 * \min\{1, 0.1 * \|\varphi(x^j)\|\}, & k < 2 \,\text{and}\, j = 1, \\ 0.008 * \min\{1, 0.1 * \|\varphi(x^j)\|\}, & k < 2 \,\text{and}\, j > 1, \\ 0.0005 * \min\{1, \|\varphi(x^j)\|\}, & \text{otherwise}. \end{cases}
$$

The parameters involved in the NW method are the same as parameters in [10]. For ADMM in Algorithm 5, if $n = 2$, we set the penalty parameter $\sigma = 4 + \sqrt{m}/90$. Otherwise, we set $\sigma = 6/n + \sqrt{m}/(810\sqrt{n})$.

In our experiments, the numbers of points $m$ are set to 1000, 2000, 5000, 10000, 20000, 50000, 100000, 200000, 500000 and the points $a_{[i]}$ $(i = 1, 2, \ldots, m)$ are generated randomly in $(-100, 100)$. The weights $w_i$ $(i = 1, 2, \ldots, m)$ are randomly generated in $(0, 100)$.

The following three tables report the comparison results on the average of 5 instances when tol $= 10^{-8}$ in iterations (iter), running time (time) and relative residuals (Res) for all the tested algorithms when $n = 2$, 5 and 10 respectively.

Table 1: The performance of SSNAL, MW, ADMM, NW when $n = 2$. In the table, "a" = SSNAL, "b" = MW, "c" = ADMM, "d" = NW. Time is shown in seconds.

| $m$ | iter | time | Res |
|---|---|---|---|
| | a\| b\| c\| d | a\| b\| c\| d | a\| b\| c\| d |
| 1e+3 | 2.0 \| 40.4 \| 183.2 \| 4.0 | 0.002 \| 0.021 \|0.027 \| 0.030 | 3.8e-10 \| 7.8e-9 \| 9.5e-9 \| 3.7e-10 |
| 2e+3 | 2.0 \| 39.0 \| 174.4 \| 4.0 | 0.003 \| 0.039 \|0.050 \| 0.105 | 2.8e-11 \| 6.7e-9 \| 9.2e-9 \| 6.2e-12 |
| 5e+3 | 2.0 \| 39.8 \| 192.8 \| 4.0 | 0.004 \| 0.092 \|0.078 \| 0.375 | 5.4e-10 \| 6.2e-9 \| 9.6e-9 \| 6.5e-12 |
| 1e+4 | 2.0 \| 38.4 \| 312.4 \| 4.0 | 0.006 \| 0.172 \|0.206 \| 1.175 | 2.7e-10 \| 6.7e-9 \| 9.8e-9 \| 1.1e-11 |
| 2e+4 | 2.0 \| 40.0 \| 861.2 \| 4.0 | 0.023 \| 0.353 \|0.906 \| 3.683 | 4.1e-11 \| 8.2e-9 \| 9.8e-9 \| 2.3e-11 |
| 5e+4 | 2.0 \| 40.0 \| 355.6 \| 4.0 | 0.041 \| 0.853 \|0.813 \| 18.523 | 7.1e-11 \| 7.4e-9 \| 9.8e-9 \| 6.9e-11 |
| 1e+5 | 2.0 \| 41.4 \| 706.4 \| 4.0 | 0.082 \| 1.763 \|4.879 \| 99.547 | 2.7e-10 \| 6.9e-9 \| 9.9e-9 \| 1.2e-10 |
| 2e+5 | 2.0 \| 41.4 \| 388.8 \| 4.0 | 0.181 \| 3.644 \|7.560 \| 594.491 | 3.4e-11 \| 7.3e-9 \| 9.7e-9 \| 2.7e-10 |
| 5e+5 | 2.0 \| 42.4 \| 655.6 \| 4.0 | 0.402 \| 9.109 \|27.773 \| 3196.520 | 3.0e-11 \| 6.7e-9 \| 9.9e-9 \| 6.4e-10 |

The numerical results of MW, NW, ADMM and SSNAL algorithms when $n = 2$ are presented in Table 1. In terms of iterations, it is observed that the SSNAL algorithm is significantly less than other algorithms. Similarly, in terms of time, the SSNAL algorithm is at least 10, 13 and 15 times faster than MW, ADMM and NW respectively. In particular, SSNAL takes 0.4 seconds to reach the required accuracy for $m =$ 5e+5, while other algorithms take about 9 to 3196 seconds. Thus, one obtains that the efficiency and stability of the SSNAL algorithm is superior to others when $n = 2$.

Table 2: The performance of Ssnal, MW, ADMM, NW when $n = 5$. In the table, "a" = Ssnal, "b" = MW, "c" = ADMM, "d" = NW. Time is shown in seconds.

| $m$ | iter | time | Res |
|---|---|---|---|
| | a\| b\| c\| d | a\| b\| c\| d | a\| b\| c\| d |
| 1e+3 | 2.0 \| 16.8 \| 58.4 \| 4.0 | 0.002 \| 0.012 \|0.012 \| 0.034 | 2.4e-11 \| 5.3e-9 \| 8.3e-9 \| 1.8e-12 |
| 2e+3 | 2.0 \| 17.0 \| 60.2 \| 4.0 | 0.003 \| 0.018 \|0.019 \| 0.109 | 2.6e-11 \| 5.8e-9 \| 8.6e-9 \| 3.2e-12 |
| 5e+3 | 2.0 \| 17.2 \| 56.6 \| 4.0 | 0.004 \| 0.041 \|0.030 \| 0.453 | 2.6-11 \| 5.0e-9 \| 9.0e-9 \| 9.3e-12 |
| 1e+4 | 2.0 \| 17.2 \| 59.4 \| 4.0 | 0.011 \| 0.079 \|0.055 \| 1.442 | 2.6e-11 \| 6.2e-9 \| 9.2e-9 \| 1.6e-11 |
| 2e+4 | 2.0 \| 17.2 \| 62.6 \| 4.0 | 0.024 \| 0.156 \|0.104 \| 5.069 | 2.7e-11 \| 5.8e-9 \| 9.0e-9 \| 3.8e-11 |
| 5e+4 | 2.0 \| 18.0 \| 76.2 \| 3.8 | 0.082 \| 0.432 \|0.574 \| 55.948 | 3.0e-11 \| 3.5e-9 \| 9.1e-9 \| 2.9e-10 |
| 1e+5 | 2.0 \| 18.0 \| 77.6 \| 3.8 | 0.122 \| 0.827 \|1.007 \| 186.529 | 2.9e-11 \| 4.4e-9 \| 8.9e-9 \| 1.4e-9 |
| 2e+5 | 2.0 \| 18.0 \| 78.2 \| 4.0 | 0.265 \| 1.702 \|2.555 \| 954.524 | 3.1e-11 \| 7.2e-9 \| 9.1e-9 \| 2.5e-10 |
| 5e+5 | 2.0 \| 18.4 \| 85.2 \| 3.6 | 0.554 \| 4.228 \|6.493 \| 5422.649 | 3.0e-11 \| 6.3e-9 \| 9.3e-9 \| 2.2e-9 |

Table 2 reports the results of all tested algorithms when $n = 5$. One can see that all tested algorithms successfully solve all instances but the running time of the Ssnal algorithm is less than others algorithms. Furthermore, the Ssnal algorithm is 6 to 10 times faster than MW, 6 to 12 times faster than ADMM and at least 17 times faster than NW. In Table 2, for the instance $m = 2\text{e} + 5$, the Ssnal algorithm only takes 0.27 seconds to produce the required accuracy solution, while the other algorithms take about 2 to 950 seconds. Finally, one can observe from the table that the larger $m$ is, the more obvious the advantage of the Ssnal algorithm in comparison with other algorithms is.

Table 3: The performance of Ssnal, MW, ADMM, NW when $n = 10$. In the table, "a" = Ssnal, "b" = MW, "c" = ADMM, "d" = NW. Time is shown in seconds.

| $m$ | iter | time | Res |
|---|---|---|---|
| | a\| b\| c\| d | a\| b\| c\| d | a\| b\| c\| d |
| 1e+3 | 2.0 \| 12.2 \| 38.6 \| 4.0 | 0.002 \| 0.008 \|0.010 \| 0.046 | 1.9e-11 \| 5.6e-9 \| 7.9e-9 \| 1.9e-12 |
| 2e+3 | 2.0 \| 12.0 \| 38.4 \| 4.0 | 0.004 \| 0.014 \|0.016 \| 0.150 | 2.1e-11 \| 4.9e-9 \| 7.6e-9 \| 3.7e-12 |
| 5e+3 | 2.0 \| 12.0 \| 37.8 \| 4.0 | 0.008 \| 0.031 \|0.028 \| 0.622 | 2.8e-11 \| 4.9e-9 \| 9.4e-9 \| 1.2e-11 |
| 1e+4 | 2.0 \| 12.0 \| 40.4 \| 4.0 | 0.017 \| 0.059 \|0.055 \| 2.145 | 3.1e-11 \| 6.6e-9 \| 8.4e-9 \| 1.9e-11 |
| 2e+4 | 2.0 \| 12.0 \| 40.0 \| 4.0 | 0.078 \| 0.137 \|0.288 \| 22.994 | 3.1e-11 \| 8.7e-9 \| 7.8e-9 \| 4.6e-11 |
| 5e+4 | 2.0 \| 13.0 \| 42.0 \| 4.0 | 0.135 \| 0.345 \|0.590 \| 99.394 | 3.2e-11 \| 1.5e-9 \| 8.6e-9 \| 8.9e-11 |
| 1e+5 | 2.0 \| 13.0 \| 40.6 \| 4.0 | 0.220 \| 0.668 \|1.088 \| 356.409 | 3.2e-11 \| 1.7e-9 \| 8.9e-9 \| 2.7e-10 |
| 2e+5 | 2.0 \| 13.0 \| 42.4 \| 4.0 | 0.425 \| 1.366 \|2.422 \| 1600.390 | 3.4e-11 \| 2.5e-9 \| 8.5e-9 \| 3.4e-10 |
| 5e+5 | 2.0 \| 13.0 \| 45.2 \| 4.0 | 0.924 \| 3.378 \|6.069 \| 9936.131 | 3.7e-11 \| 3.7e-9 \| 9.1e-9 \| 9.0e-10 |

Table 3 presents the performance of all tested algorithms when $n = 10$. We observe that all tested algorithms still succeed in solving all instances to the desired accuracy. Careful comparison shows that the Ssnal algorithm is about 4 times faster than MW, about 5 times faster than ADMM and at least 23 times faster than NW. For the instance $m = 5\mathrm{e}+5$, the Ssnal algorithm solves it within 1 second, while MW and ADMM take 3.38 and 6.07 seconds to solve this instance, and the NW algorithm even takes 9936.13 seconds which is close to the maximum running time we set. Therefore, the Ssnal algorithm also outperforms other algorithms when $n = 10$.

Consequently, we can safely claim that the Ssnal algorithm substantially outperforms MW, NW and ADMM in terms of efficiency and robustness.

## 5 Conclusion

In this paper, we have developed a highly efficient semismooth Newton based augmented Lagrangian algorithm for the Weber problem (1.1). The theoretical results showed that the Ssnal algorithm admits global convergence and locally asymptotically superlinear convergence. Moreover, numerical experiments demonstrated that the Ssnal algorithm is superior to MW, ADMM and NW algorithms in terms of running time and robustness.

## References


[1] A. Beck, *First-order Methods in Optimization*, SIAM, Philadelphia, 2017.

[2] A. Beck and S. Sabach, Weiszfeld's method: old and new results, *J. Optim. Theory Appl.* 164 (2015) 1–40.

[3] F.H. Clarke, *Optimization and Nonsmooth Analysis*, SIAM, Philadelphia, 1990.

[4] Y. Cui, C. Ding and X. Y. Zhao, Quadratic growth conditions for convex matrix optimization problems associated with spectral functions, *SIAM J. Optim.* 27 (2017) 2332–2355.

[5] Y. Cui, D. F. Sun and K.-C. Toh, On the R-superlinear convergence of the KKT residues generated by the augmented Lagrangian method for convex composite conic programming, *Math. Program.* 178 (2019) 381–415.

[6] A.L. Dontchev and R.T. Rockafellar, *Implicit Functions and Solution Mappings*, Springer, New York, 2009.

[7] Z. Drezner, *Facility Location: A Survey of Applications and Methods*, Springer, Berlin, 1995.

[8] Z. Drezner and H.W. Hamacher, *Facility Location: Applications and Theory*, Springer, Berlin, 2002.

[9] R.Z. Farahani and M. Hekmatfar, *Facility Location: Concepts, Models, Algorithms and Case Studies*, Springer, New York, 2009.

[10] S. Görner and C. Kanzow, On Newton's method for the Fermat-Weber location problem, *J. Optim. Theory Appl.* 170 (2016) 107–118.

[11] T. Jahn, Y. S. Kupitz, H. Martini and C. Richter, Minsum location extended to gauges and to convex sets, *J. Optim. Theory Appl.* 166 (2015) 711–746.

[12] J.L. Jiang, K. Cheng, C.C. Wang and L.P. Wang, Accelerating the convergence in the single-source and multi-source Weber problems, *Appl. Math. Comput.* 218 (2012) 6814–6824.

[13] J.L. Jiang and X.M. Yuan, A heuristic algorithm for constrained multi-source Weber problem-The variational inequality approach, *European J. Oper. Res.* 187 (2008) 357–370.

[14] I.N. Katz, Local convergence in Fermat's problem, *Math. Program.* 6 (1974) 89–104.

[15] B. Kummer, Newton's method for non-differentiable functions, *Adv. Math. Optim.* 45 (1988) 114–125.

[16] B. Kummer, Newton's method based on generalized derivatives for nonsmooth functions: convergence analysis, in *Advances in Optimization*, Springer, 1992 pp. 171–194.

[17] Y.S. Kupitz, H. Martini and M. Spirova, The Fermat-Torricelli problem, part I: A discrete gradient-method approach, *J. Optim. Theory Appl.* 158 (2013) 305–327.

[18] C. Lemaréchal and C. Sagastizábal, Practical aspects of the Moreau-Yosida regularization: theoretical preliminaries, *SIAM J. Optim.* 7 (1997) 367–385.

[19] Y. Levin and A. Ben-Israel, The Newton bracketing method for convex minimization, *Comput. Optim. Appl.* 21 (2002) 213–229.

[20] X. D. Li, D.F. Sun and K.-C. Toh, A highly efficient semismooth Newton augmented Lagrangian method for solving Lasso problems, *SIAM J. Optim.* 28 (2018) 433–458.

[21] X.D. Li, D.F Sun and K.-C. Toh, On efficiently solving the subproblems of a level-set method for fused Lasso problems, *SIAM J. Optim.* 28 (2018) 1842–1866.

[22] Y.Y. Li, A Newton acceleration of the Weiszfeld algorithm for minimizing the sum of euclidean distances, *Comput. Optim. Appl.* 10 (1998) 219–242.

[23] L.Y. Lin and Y.-J. Liu, An efficient Hessian based algorithm for singly Linearly and box constrained least squares regression, *J. Sci. Comput.* 88 (2021) 1–21.

[24] M.X. Lin, Y.-J. Liu, D.F. Sun and K.-C. Toh, Efficient sparse semismooth Newton methods for the clustered Lasso problem, *SIAM J. Optim.* 29 (2019) 2026–2052.

[25] R.F. Love and J.G. Morris, Mathematical models of road travel distances, *Manage. Sci.* 25 (1979) 130–139.

[26] R.F. Love, J.G. Morris and G.O. Wesolowsky, *Facilities Location: Models and Methods*, Elsevier, New York, 1988.

[27] Z.Y. Luo, D.F Sun, K.-C. Toh and N.H. Xiu, Solving the OSCAR and SLOPE models using a semismooth Newton-based augmented Lagrangian method, *J. Mach. Learn. Res.* 20 (2019) 1–25.

[28] H. Martini, K.J. Swanepoel and G. Weiss, The Fermat-Torricelli problem in normed planes and spaces, *J. Optim. Theory Appl.* 115 (2002) 283–314.

[29] B. Mordukhovich and N.M. Nam, Applications of variational analysis to a generalized Fermat-Torricelli problem, *J. Optim. Theory Appl.* 148 (2011) 431–454.

[30] J.-J. Moreau, Proximité et dualité dans un espace hilbertien, *Bull. Soc. Math. France* 93 (1965) 273–299.

[31] S. Nickel and J. Puerto, *Location Theory: A Unified Approach*, Springer, Berlin, 2006.

[32] D.B. Niu, C.J. Wang, P.P. Tang, Q.S. Wang and E.B. Song, A sparse semismooth Newton based augmented Lagrangian method for large-scale support vector machines, *arXiv preprint arXiv:1910.01312*, 2019.

[33] L.Q. Qi and J. Sun, A nonsmooth version of Newton's method, *Math. Program.* 58 (1993) 353–367.

[34] R.T. Rockafellar. *Convex Analysis*, Princeton Univ. Press, Princeton, 1970.

[35] R.T. Rockafellar, Augmented Lagrangians and applications of the proximal point algorithm in convex programming, *Math. Oper. Res.* 1 (1976) 97–116.

[36] R.T. Rockafellar, Monotone operators and the proximal point algorithm, *SIAM J. Control Optim.* 14 (1976) 877–898.

[37] R.T. Rockafellar and R.J.-B. Wets, *Variational Analysis*, Springer, Berlin, 1998.

[38] Y. Vardi and C.H. Zhang, A modified Weiszfeld algorithm for the Fermat-Weber location problem, *Math. Program.* 90 (2001) 559–566.

[39] E. Weiszfeld, Sur le point pour lequal la somme des distance de n points donnés est minimum, *Tohoku Math. J.* 43 (1937) 355–386.

[40] S.L. Yan and J.L. Jiang, New numerical methods for Weber problem based on alternating direction method of multipliers, *J. Henan Univ. Nat. Sci.* 48 (2018) 740–750.

[41] Y.J. Zhang, N. Zhang, D.F. Sun and K.-C. Toh, An efficient Hessian based algorithm for solving large-scale sparse group Lasso problems, *Math. Program.* 179 (2020) 223–263.

[42] X.Y. Zhao, D.F Sun and K.-C. Toh, A Newton-CG augmented Lagrangian method for semidefinite programming, *SIAM J. Optim.* 20 (2010) 1737–1765.

---

*Manuscript received 28 June 2021*
*revised 12 September 2021*
*accepted for publication 7 October 2021*

Yong-Jin Liu
School of Mathematics and Statistics, Fuzhou University
Fuzhou, 350108, China
E-mail address: yjliu@fzu.edu.cn

Qinxin Zhu
School of Mathematics and Statistics, Fuzhou University
Fuzhou, 350108, China
E-mail address: qxzhu1998@163.com