NEWTON-TYPE METHODS FOR THE FERMAT-WEBER PROBLEM WITH WEIGHTED EUCLIDEAN NORMS

BENJAMIN PAUL-DUBOIS-TAINE

ABSTRACT. This report studies Newton-type methods for solving the Fermat-Weber. It extends the work by Görner and Kanzow (JOTA 170(1), 2016, pp. 107–118) in that it allows for weighted Euclidean norms, and also considers the box-constrained case. Theoretical results for well-definedness and convergence are provided, and numerical tests are presented, comparing the studied methods with standard algorithms such as FISTA.

1 Introduction

The Fermat-Weber problem consists in finding a point that minimizes the sum of weighted distances to a given finite set of points. It reads

$$\min_{x\in\mathbb{R}^n} f(x) := \sum_{i=1}^{m} \left\|x - a^i\right\|$$

where the points a^i are given. It has been extensively studied in many different settings, e.g by working with different distance measures, see [1] for references. This paper studies the case of the weighted Euclidean norms.
Historically, the method used to solve find the minimizer of the Fermat-Weber with Euclidean distance has been Weiszfeld's algorithm, a fixed-point iteration method. Although widely used, this algorithm has slow convergence, as it can be interpreted as a gradient-type method with linear convergence, see for example [2].
In recent research, it was found that after a suitable initialization, Newton's method can be applied to the Fermat-Weber problem with Euclidean norm, leading to a quadratic rate of convergence, see [1].
We show that this result can be generalized to the Fermat-Weber problem with what we call the weighted Euclidean norm, leading to the same convergence rates. Moreover, we also show that a similar initialization can be done for the Fermat-Weber with box constraints, leading to the possible use of Newton-type methods.

The paper is organized as follows: In Section 2, we recall some known facts about the Fermat-Weber problem and propose an approach to find a starting point for Newton-type methods. In Section 3, we study the unconstrained Fermat-Weber and show that Newton's method can be used and yields quadratic convergence. Finally, in Section 4, we study the problem with box constraints, which leads to the use of the projected Newton method. We provide numerical evidence to support the theoretical results.

Key words and phrases. Fermat-Weber problem, Newton's method, quadratic convergence, superlinear convergence, convex optimization, subdifferential.

2 Preliminaries

For a function $f : \mathbb{R}^n \to \mathbb{R}$ we say that f is *convex* if

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \quad (x, y \in \mathbb{R}^n, \lambda \in (0,1)).$$

The *sublevel sets* of f are defined as

$$\mathcal{L}_f(c) := \{x \in \mathbb{R}^n \mid f(x) \leq c\} \quad (c \in \mathbb{R}^n).$$

We say that f is *level-bounded* if $\mathcal{L}_f(c)$ is bounded for all $c \in \mathbb{R}^n$.
For a convex function f, the *subdifferential* at $x \in \mathbb{R}^n$ is defined as

$$\partial f(x) := \left\{v \in \mathbb{R}^n \;\middle|\; f(y) - f(x) \geq v^T(y-x) \quad (y \in \mathbb{R}^n)\right\}$$

We would also like to introduce the notion of *weighted Euclidean norms*. For the remaining of the paper, we let $H \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix. The following is then an inner product on $\mathbb{R}^n$

$$\langle x, y\rangle_H := x^T H y \quad (x, y \in \mathbb{R}^n)$$

The norm induced by this inner is what we call a weighted Euclidean norm, i.e.

$$\|x\|_H := \sqrt{\langle x, x\rangle_H} = \sqrt{x^T H x}$$

The case of $H = I$, the identity matrix, gives the usual Euclidean norm. Finally, observe that since H is symmetric positive definite, there exists a unique symmetric positive definite matrix $S \in \mathbb{R}^{n\times n}$ such that $S^2 = H$. Moreover, we have

$$\|x\|_H = \sqrt{x^T H x} = \sqrt{x^T S S x} = \sqrt{\langle Sx, Sx\rangle_I} = \|Sx\|_I \quad (x \in \mathbb{R}^n)$$

This property will be very useful later on.

With these definitions in mind, we are now ready to introduce the problem. Mathematically, the Fermat-Weber problem reads

$$\min f(x) := \sum_{i=1}^{m} \omega_i ||x - a^i|| \quad \text{subject to } x \in X \tag{1}$$

where the vectors $a^1, \ldots, a^m \in \mathbb{R}^n$ denote pairwise disjoint points, sometimes called anchor points, X is a nonempty closed convex set, the scalars $\omega_i > 0$ are positive weights, and $||\cdot||$ stands for some norm on $\mathbb{R}^n$. We assume that the anchor points are not collinear and that $m \geq 3$.
We start with a few properties of the Fermat-Weber problem, without any assumption on the norm.

Proposition 2.1. *The following statements hold:*
(a) The function f from from (1) is convex
(b) The problem (1) always has a solution

Proof. (a) f is convex as a sum of convex functions, i.e. the functions

$$x \mapsto \omega_i \left\|x - a^i\right\| \quad (i = 1, \ldots, m)$$

The summands are, in fact, convex as positive multiples of a convex function (the norm function) composed with an affine map, see [9, prop 2.1.1] and [9, prop 2.1.4].
(b) f is clearly level-bounded and continuous. Therefore its sublevel sets are closed. Because X is also closed, this implies that f takes its minimum over X (by Weierstrass' theorem). □

From now on, we write $\|\cdot\|$ for the usual Euclidean norm, i.e. $\|x\| := \|x\|_I$. Now, we would like to study the Fermat-Weber in the more specific setting of the weighted Euclidean norm. The Fermat-Weber problem then reads

$$\min f(x) := \sum_{i=1}^{m} \omega_i ||x - a^i||_H \quad \text{subject to } x \in X \tag{2}$$

The case of $H = I$ and $X = \mathbb{R}^n$ is the focus of [1]. In this paper, the authors show how, after a suitable initialization, Newton's method can be applied to the problem to achieve quadratic convergence. In this section we show how a similar initialization can be done in the more general setting of (2).
First, let's point out an interesting of the problem (2).

Proposition 2.2. *The solution of (2) is unique.*

Proof. It suffices to show that f is strictly convex. Suppose this were not the case. Then there exists $x, y \in \mathbb{R}^n$, $x \neq y$ and $\lambda \in (0,1)$ such that

$$f(\lambda x + (1-\lambda)y) = \lambda f(x) + (1-\lambda) f(y)$$

This means that

$$\begin{aligned}\sum_{i=1}^{m} \omega_i ||\lambda(x-a^i) + (1-\lambda)(y-a^i)||_H &= \lambda \sum_{i=1}^{m} \omega_i ||(x-a^i)||_H + (1-\lambda)\sum_{i=1}^{m} \omega_i ||(y-a^i)||_H \\ &= \sum_{i=1}^{m} \omega_i \lambda ||(x-a^i)||_H + \omega_i(1-\lambda)||(y-a^i)||_H\end{aligned}$$

For this equality to hold, it is necessary that all summands be equal (because of the triangle inequality property of the norm), i.e. we have

$$||\lambda(x-a^i) + (1-\lambda)(y-a^i)||_H = \lambda ||x-a^i||_H + (1-\lambda)||y-a^i||_H \quad (i = 1,\dots,m)$$

which we can rewrite

$$||S(\lambda(x-a^i) + (1-\lambda)(y-a^i))|| = \lambda ||S(x-a^i)|| + (1-\lambda)||S(y-a^i)|| \quad (i = 1,\dots,m)$$

We are here in the case of equality of the triangle inequality for the 2-norm. Hence we have that $S(x-a^i)$ and $S(y-a^i)$ are linearly dependent. Hence there exists $\alpha_i \in \mathbb{R}$ such that

$$S(x-a^i) = \alpha_i S(y-a^i)$$

which implies, since S is invertible,

$$x - a^i = \alpha_i (y - a^i)$$

Note that $\alpha_i \neq 1$ since $x \neq y$. Hence we get:

$$\begin{aligned}a^i &= \frac{\alpha_i}{\alpha_i - 1} y - \frac{1}{\alpha_i - 1} x \\ &= \frac{\alpha_i}{\alpha_i - 1} y + (1 - \frac{\alpha_i}{\alpha_i - 1}) x\end{aligned}$$

This implies that all the anchor points a^i are on the same line passing the points x and y. But this contradicts the assumption that the anchor points are not collinear. Hence f is strictly convex which concludes the proof. $\square$

To apply Newton-type methods, the underlying function needs to be (twice) differentiable at each iteration. The function f from (2) is differentiable on $\mathbb{R}^n \setminus \{a^1, \dots, a^m\}$. As in [1], the approach is as follows: first check a criterion for the anchor point with smallest function to be a solution of (2). If it is not, find a descent direction at that point and start the algorithm at a point with smaller function value. As long as the method is guaranteed to decrease the function value, a point of non-differentiability will never be reached.
Before we present the criterion, we need one last assumption.

Assumption 2.3. *No anchor point is on the boundary of X, i.e.*

$$a^i \in \operatorname{int}(X) \text{ or } a^i \notin X \quad (i = 1, \dots, m)$$

Proposition 2.4. *Consider the problem (2) and assume that Assumption 2.3 holds. Determine $p \in \{1, \dots, m\}$ such that $f(a^p) = \min_{i=1,\dots,m} f(a^i)$.*
(a) If $a^p \notin X$, then the solution of (2) is not an anchor point.
(b) If $a^p \in X$, then a^p solves (2) if and only if

$$\left\| \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| \leq \omega_p$$

Proof. (a) By contradiction, assume that a^j solves (2) for some $a^j \in X$. Then, as $a^j \in \operatorname{int}(X)$ by Assumption 2.3, we have that a^j is a local minimizer of f. Since f is convex, this implies that a^j is a global minimizer of f. By strict convexity of f, the global minimizer is unique. But then $f(a^p) > f(a^j)$ which is a contradiction.

(b) If $a^p \in X$, then since $a^p \in \operatorname{int}(X)$, we have by [9, Theorem 2.2.1],

$$a^p \in \arg\min_X f \iff 0 \in \partial f(a^p) \tag{3}$$

Writing

$$g_i(x) = \omega_i \left\| x - a^i \right\|_H = \omega_i \left\| S(x - a^i) \right\|$$

we get

$$f(x) = \sum_{i=1}^{m} g_i(x)$$

Now, a short computation shows that

$$\partial g_i(x) = \begin{cases} \left\{ \omega_i \frac{H(x-a^i)}{\|S(x-a^i)\|} \right\}, & \text{if } x \neq a^i \\ \\ \omega_i S\mathbb{B}, & \text{if } x = a^i \end{cases}$$

where $\mathbb{B} = \{ y \in \mathbb{R}^n | \, \|y\| \leq 1 \}$.
Since the subdifferential of a sum of finite-valued convex functions is simply the sum of the subdifferentials of the functions ([9, Theorem 4.1.1]), we get that

$$\partial f(a^p) = \left\{ \omega_p S y + \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|} \quad | \quad \|y\| \leq 1 \right\}$$

Now, suppose $0 \in \partial f(a^p)$. Then there exists $y \in \mathbb{R}^n$ such that $\|y\| \le 1$ and

$$0 = \omega_p S y + \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|}$$

Since S is positive definite, it is invertible and hence multiplying by S^{-1} we get

$$0 = \omega_p y + \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|}$$

Then

$$\begin{aligned}\left\| \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| &= \|\omega_p y\| \\ &= \omega_p \|y\| \\ &\le \omega_p\end{aligned}$$

Now we can prove the other direction. Suppose we have

$$\left\| \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| \le \omega_p$$

Set

$$y = -\frac{1}{\omega_p} \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|}$$

Then $\|y\| \le 1$ and we have

$$\begin{aligned}\sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|} + \omega_p S y &= \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|} - \omega_p S \frac{1}{\omega_p} \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \\ &= \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|} - \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(a^p - a^i)}{\|S(a^p - a^i)\|} \\ &= 0\end{aligned}$$

Hence $0 \in \partial f(a^p)$. We just proved that

$$0 \in \partial f(a^p) \iff \left\| \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| \le \omega_p$$

By (3), we are done.

□

Since f is convex, the directional derivative exists for all $x \in \mathbb{R}^n$ and in all direction $d \in \mathbb{R}^n$. Denote

$$K := \left\{ i \in \{1, \ldots, m\} \quad | \quad a^i \in X \right\}$$

Now suppose that we have a^p such that $f(a^p) = \min_{i \in K} f(a^i)$ but we determined by Proposition 2.4 that a^p is not a minimizer of f over X. Then there exists a descent direction of f at a^p. One way to find one is to solve the following optimization problem

$$\min_d f'(a^p, d) \quad \text{s.t.} \quad \|d\|_H = 1 \tag{4}$$

Proposition 2.5. *The vector*

$$d_p := \frac{-R_p}{\|R_p\|_H} \quad \text{with} \quad R_p := \sum_{i=1, i\neq p}^{m} \omega_i \frac{a^p - a^i}{\|a^p - a^i\|_H}$$

is a solution of (4).

Proof. First, observe that we can write:

$$f(x) = \omega_p \|x - a^p\|_H + f_p(x)$$

where

$$f_p(x) = \sum_{i=1, i\neq p}^{m} \omega_i \left\|x - a^i\right\|_H$$

Now, taking $d \in \mathbb{R}^n$ satisfying $\|d\|_H = 1$, we can define

$$\alpha(t) = f(a^p + td) = \omega_p t \|d\|_H + f_p(a^p + td) = \omega_p t + f_p(a^p + td)$$

Observe that f_p is differentiable at a^p. Then the directional derivative of f at a^p in the direction of d is given by

$$f'(a^p, d) = \alpha'(0) = \omega_p + \nabla f_p(a^p)^T d$$

We would like to minimize this quantity. Observe that for all d such that $\|d\|_H = 1$, we have

$$\begin{aligned}
\nabla f_p(a^p)^T d &= \nabla f_p(a^p)^T H^{-1} H d \\
&= \langle H^{-1} \nabla f_p(a^p), d\rangle_H \\
&\geq -\left\|H^{-1}\nabla f_p(a^p)\right\|_H \|d\|_H \quad \text{by C-S inequality} \\
&= -\left\|H^{-1}\nabla f_p(a^p)\right\|_H
\end{aligned}$$

Now, setting

$$d_p := -\frac{H^{-1}\nabla f_p(a^p)}{\|H^{-1}\nabla f_p(a^p)\|_H}$$

we get

$$\begin{aligned}
\nabla f_p(a^p)^T d_p &= -\frac{\nabla f_p(a^p)^T H^{-1} \nabla f_p(a^p)}{\|H^{-1}\nabla f_p(a^p)\|_H} \\
&= -\frac{(H^{-1}\nabla f_p(a^p))^T H H^{-1} \nabla f_p(a^p)}{\|H^{-1}\nabla f_p(a^p)\|_H} \\
&= -\frac{\left\|H^{-1}\nabla f_p(a^p)\right\|_H^2}{\left\|H^{-1}\nabla f_p(a^p)\right\|_H} = -\left\|H^{-1}\nabla f_p(a^p)\right\|_H
\end{aligned}$$

Hence the smallest directional derivative is attained at d_p. Finally, observe that

$$\nabla f_p(x) = \sum_{i=1, i\neq p}^{m} \omega_i \frac{H(x - a^i)}{\|x - a^i\|_H} \quad \textit{for} \quad x \neq a^1, ..., a^{p-1}, a^{p+1}, ..., a^m$$

Hence, setting $R_p = H^{-1}\nabla f_p(a^p)$, we get the result. □

Since a_p is not a minimum of f, we have that $f'(a^p, d_p) < 0$. Hence $f(a^p + td_p) < f(a^p)$ for all $t > 0$ sufficiently small. Moreover, $a^p + td_p \in X$ for all $t > 0$ sufficiently small as $a^p \in \text{int}(X)$. Such a step-size can be found by simple backtracking. Once we find such a step-size, we can apply any algorithm that guarantees a decrease in function value, since a point of non differentiability will then never be reached. This is the focus of the next sections, where we first study the unconstrained case, followed by the case of box constraints.

3 The Unconstrained Case

We now consider the case $X = \mathbb{R}^n$. Observe that Assumption 2.3 is trivially satisfied. The algorithm works as follows. First, we check whether the criterion from Proposition 2.4 (b) holds. If it is satisfied then we are done. If not, we determine a descent direction and a step-size and start Newton's method from the new generated point x^0.

Algorithm 1 Newton's method for the Fermat-Weber problem

(S0) Determine $p \in \{1, ..., m\}$ such that $f(a^p) = \min\{f(a^1), ..., f(a^m)\}$. If a^p is such that

$$\left\| \sum_{i=1, i\neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| \leq \omega_p$$

then a^p is the minimum. STOP.

(S1) Compute

$$d_p := \frac{-R_p}{\|R_p\|_H} \quad \text{with} \quad R_p := \sum_{i=1, i\neq p}^{m} \omega_i \frac{a^p - a^i}{\|a^p - a^i\|_H}$$

and find a step-size $t_p > 0$ such that $f(a^p + t_p d_p) < f(a_p)$ by backtracking. Set $x^0 := a^p + t_p d_p$, $k := 0$, and choose parameter $\epsilon > 0$, $\rho \in (0, 1)$, $\sigma \in (0, 1/2)$.

(S2) If $\|\nabla f(x^k)\| \leq \epsilon$, then STOP.

(S3) Compute the Newton direction d^k by solving $\nabla^2 f(x^k) d = -\nabla f(x^k)$.

(S4) Compute a step-size t_k as the largest number in $\{1, \rho, \rho^2, ...\}$ such that

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

(S5) Set $x^{k+1} := x^k + t_k d^k, k \leftarrow k + 1$, and go to (S2).

From now on, we refer to x^0 as the starting point of Algorithm 1, as defined in (S1).
We begin our convergence analysis by proving that the Hessians of f are positive definite on the level set $\mathcal{L}_f(x^0)$, which we write $\mathcal{L}(x^0)$ for convenience. This guarantees that the steps (S3) and (S4) of the algorithm are well defined.

Proposition 3.1. *For any $x \in \mathcal{L}(x^0)$, the Hessian $\nabla^2 f(x)$ is positive definite.*

Proof. First observe that f is twice continuously differentiable on an open set containing the level set $\mathcal{L}(x^0)$. This is because f is twice continuously differentiable on $\mathbb{R}^n \setminus \{a^1, ..., a^m\}$ and for all $x \in \mathcal{L}(x^0)$,

$$f(x) \leq f(x^0) < \min_{i=1,\dots,m} f(a^i)$$

and hence $x \notin \{a^1, ..., a^m\}$.

Since f is convex, we have that the Hessian $\nabla^2 f(x)$ is positive semidefinite for all $x \in \mathcal{L}(x^0)$. Moreover, a quick calculation shows that

$$\nabla^2 f(x) = \sum_{i=1}^{m} \frac{\omega_i}{\|x - a^i\|_H^3} \Big(\|x - a^i\|_H^2 H - H(x - a^i)(x - a^i)^T H \Big)$$

Using the Cauchy-Schwartz inequality

$$\begin{aligned} d^T \nabla^2 f(x) d &= \sum_{i=1}^{m} \frac{\omega_i}{\|x - a^i\|_H^3} \Big(\|x - a^i\|_H^2 d^T H d - d^T H(x - a^i)(x - a^i)^T H d \Big) \\ &= \sum_{i=1}^{m} \frac{\omega_i}{\|x - a^i\|_H^3} \Big(\|x - a^i\|_H^2 \|d\|_H^2 - (d^T H(x - a^i))^2 \Big) \\ &= \sum_{i=1}^{m} \frac{\omega_i}{\|x - a^i\|_H^3} \Big(\|x - a^i\|_H^2 \|d\|_H^2 - \langle d, x - a^i \rangle_H^2 \Big) \\ &\geq \sum_{i=1}^{m} \frac{\omega_i}{\|x - a^i\|_H^3} \Big(\|x - a^i\|_H^2 \|d\|_H^2 - \|x - a^i\|_H^2 \|d\|_H^2 \Big) \\ &= 0 \end{aligned}$$

for all $d \neq 0$ and equality holds only if the vectors d and $x - a^i$ are linearly dependent $(i = 1, ..., m)$. But that would imply that all the points $a^1, ..., a^m$ are on the same line which is a contradiction. Hence we have strict inequality and therefore the Hessians are positive definite. □

Corollary 3.2. *There exist constants $\beta \geq \alpha > 0$ such that*

$$\alpha \|d\|^2 \leq d^T \nabla^2 f(x) d \leq \beta \|d\|^2$$

for all $d \in \mathbb{R}^n$ and $x \in \mathcal{L}(x^0)$.

Proof. Denote by $\lambda_{max}(x)$ and $\lambda_{min}(x)$ the largest and smallest eigenvalues, respectively, of $\nabla^2 f(x)$, for $x \in \mathcal{L}(x^0)$. It is easy to show that

$$\lambda_{min}(x) d^T d \leq d^T \nabla^2 f(x) d \leq \lambda_{max}(x) d^T d$$

Now, observe that the mappings that associate x to $\lambda_{min}(x)$ and $\lambda_{max}(x)$ are continuous functions on $\mathcal{L}(x^0)$. Moreover, $\mathcal{L}(x^0)$ is bounded and closed (since f is continuous), i.e. compact. Hence $\lambda_{min}(.)$ and $\lambda_{max}(.)$ reach their infimum and supremum on $\mathcal{L}(x^0)$. Write

$$\lambda_{min} := \min_{x \in \mathcal{L}(x^0)} \lambda_{min}(x) \quad \text{and} \quad \lambda_{max} := \max_{x \in \mathcal{L}(x^0)} \lambda_{max}(x)$$

Then we get

$$\lambda_{min} d^T d \leq d^T \nabla^2 f(x) d \leq \lambda_{max} d^T d \quad (x \in \mathcal{L}(x^0))$$

Which we can rewrite

$$\lambda_{min} \|d\|^2 \leq d^T \nabla^2 f(x) d \leq \lambda_{max} \|d\|^2 \quad (x \in \mathcal{L}(x^0))$$

Setting $\alpha := \lambda_{min}$ and $\beta := \lambda_{max}$, we are done.

□

Observe that our algorithm uses the Armijo rule to find the step-size. This step-size is not usually efficient in the sense of [10], but we show that in our case it is.

Lemma 3.3. *There exists a constant* $\theta > 0$ *such that*

$$f(x^k + t_k d^k) \leq f(x^k) - \theta\left(\frac{\nabla f(x^k)^T d^k}{\|d^k\|}\right)^2 \quad (k \in \mathbb{N}).$$

Proof. Recall that $\nabla^2 f(x^k)d^k = -\nabla f(x^k)$ for all $k \in \mathbb{N}$. Hence, using Corollary 3.2, we get

$$-\frac{\nabla f(x^k)^T d^k}{\|d^k\|^2} = \frac{(d^k)^T \nabla^2 f(x_k) d^k}{\|d^k\|^2} \leq \beta \quad (k \in \mathbb{N}) \tag{5}$$

. Now, consider $k \in \mathbb{N}$. Define $\varphi_k(t) := f(x^k + td^k)$. Since f is continuous and bounded from below, there exists a smallest $\hat{t}_k > 0$ such that

$$\varphi_k'(\hat{t}_k) = \sigma\varphi_k'(0).$$

Now, observe that ∇f is continuously differentiable on $\mathcal{L}(x^0)$ which implies that ∇f is locally Lipschitz on $\mathcal{L}(x^0)$. Since $\mathcal{L}(x^0)$ is compact, we actually have that ∇f is Lipschitz continuous on $\mathcal{L}(x^0)$.
Moreover, $x^k + td^k \in \mathcal{L}(x^0)$ for all t such that $0 < t \leq \hat{t}_k$, there exists a constant $L > 0$ satisfying

$$\begin{aligned}
\sigma\varphi_k'(0) &= \varphi_k'(\hat{t}_k)\\
&= \varphi_k'(0) + (\varphi_k'(\hat{t}_k) - \varphi_k'(0))\\
&= \varphi_k'(0) + (\nabla f(x^k + \hat{t}_k d^k)^T d^k - \nabla f(x^k)^T d^k)\\
&= \varphi_k'(0) + (\nabla f(x^k + \hat{t}_k d^k) - \nabla f(x^k))^T d^k\\
&\leq \varphi_k'(0) + \left\|\nabla f(x^k + \hat{t}_k d^k) - \nabla f(x^k)\right\|_2 \left\|d^k\right\|_2 \quad \text{by Cauchy-Schwartz inequality}\\
&\leq \varphi_k'(0) + L\left\|\hat{t}_k d^k\right\| \left\|d^k\right\|\\
&= \varphi_k'(0) + \hat{t}_k L \left\|d^k\right\|^2
\end{aligned}$$

Which implies

$$\hat{t}_k \geq -\frac{(1-\sigma)\varphi_k'(0)}{L\left\|d^k\right\|^2} \tag{6}$$

We now distinguish two cases:
<u>Case 1</u>: $t_k = 1$ in the Armijo rule. Then (5) implies

$$t_k = 1 \geq -\frac{1}{\beta}\frac{\nabla f(x^k)^T d^k}{\|d^k\|^2}$$

and therefore we get

$$\begin{aligned} f(x^k + t_k d^k) &\leq f(x^k) + \sigma t_k \nabla f(x^k)^T d^k \\ &\leq f(x^k) - \sigma \frac{\nabla f(x^k)^T d^k}{\beta \left\| d^k \right\|^2} \nabla f(x^k)^T d^k \quad (\text{since} \quad \nabla f(x^k)^T d^k < 0) \\ &= f(x^k) - \frac{\sigma}{\beta} \left(\frac{\nabla f(x^k)^T d^k}{\|d^k\|} \right)^2 \end{aligned}$$

Case 2: $t_k < 1$ in the Armijo rule. Then t_k/ρ violates the Armijo condition. On the other hand, $\hat{t}_k$ and all step-sizes $0 < t \leq \hat{t}_k$ satisfy the Armijo condition; hence, it follows that

$$\frac{1}{\rho} t_k > \hat{t}_k$$

Using (6), we have (again observe that $\nabla f(x^k)^T d^k < 0$)

$$\begin{aligned} f(x^k + t_k d^k) &\leq f(x^k) + \sigma t_k \nabla f(x^k)^T d^k \\ &< f(x^k) + \sigma \rho \hat{t}_k \nabla f(x^k)^T d^k \\ &\leq f(x^k) - \sigma \rho \frac{(1-\sigma)\varphi_k'(0)}{L \left\| d^k \right\|^2} \nabla f(x^k)^T d^k \\ &= f(x^k) - \frac{\sigma \rho (1-\sigma)}{L} \left(\frac{\nabla f(x^k)^T d^k}{\|d^k\|} \right)^2 \end{aligned}$$

Taking $\theta := \min\left\{ \frac{\sigma}{\beta}, \frac{\sigma\rho(1-\sigma)}{L} \right\}$, we are done. □

Theorem 3.4. *If Algorithm 1 does not terminate in step (S.0), then the sequence* $\left\{ x^k \right\}$ *generated by this method converges to the unique solution* x^* *of the Fermat-Weber problem (2). Furthermore, the local rate of convergence is quadratic, i.e. there is a constant* $c > 0$ *such that* $\left\| x^{k+1} - x^* \right\| \leq c \left\| x^k - x^* \right\|^2$ *for all* $k \in \mathbb{N}$ *sufficiently large.*

Proof. First, note that Corollary 3.2 gives

$$-\frac{\nabla f(x^k)^T d^k}{\|\nabla f(x^k)\|_2 \|d^k\|_2} = \frac{(d^k)^T \nabla^2 f(x^k) d^k}{\|\nabla^2 f(x^k) d^k\|_2 \|d^k\|_2} \geq \frac{(d^k)^T \nabla^2 f(x^k) d^k}{\|\nabla^2 f(x^k)\|_2 \|d^k\|_2^2} \geq \frac{\alpha}{\beta} > 0$$

for all $k \in \mathbb{N}$, assuming $\left\| \nabla^2 f(x^k) \right\|_2 \leq \beta$ holds (Note that this assumption is acceptable since $\nabla^2 f$ is continuous on the compact set $\mathcal{L}(x^0)$, and hence bounded). Hence, the search directions satisfy an angle condition and the step-sizes are efficient by Lemma 3.3. Using some standard results from [3], we get that the sequence $\left\{ x^k \right\}$ converges to x^*.
Moreover, using Taylor's theorem, we have that for all $k \in \mathbb{N}$, there exists $\mu^k \in \mathbb{R}^n$ between x^k and $x^k + d^k$ such that

$$f(x^k + d^k) = f(x^k) + \nabla f(x^k)^T d^k + \frac{1}{2} (d^k)^T \nabla^2 f(\mu^k) d^k$$

which we can rewrite as (since $\nabla f(x^k)^T d^k \neq 0$ for all k)

$$\frac{f(x^k + d^k) - f(x^k)}{\nabla f(x^k)^T d^k} = 1 + \frac{1}{2} \frac{(d^k)^T \nabla^2 f(\mu^k) d^k}{\nabla f(x^k)^T d^k}$$

Now,

$$\frac{1}{2}\frac{(d^k)^T\nabla^2 f(\mu^k)d^k}{\nabla f(x^k)^T d^k} = -\frac{1}{2}\frac{(d^k)^T\nabla^2 f(\mu^k)d^k}{(d^k)^T\nabla^2 f(x^k)d^k}$$

Recall that $\nabla^2 f(x^k)d^k = -\nabla f(x^k)$. Moreover, $\lim_{k\to\infty}\nabla f(x^k) = 0$ and the Hessian $\nabla^2 f$ is positive definite at x^*, hence we must have $\lim_{k\to\infty} d^k = 0$. But then this implies that $\lim_{k\to\infty}\mu^k = \lim_{k\to\infty} x^k = x^*$. Hence we get

$$\lim_{k\to\infty}\frac{1}{2}\frac{(d^k)^T\nabla^2 f(\mu^k)d^k}{\nabla f(x^k)^T d^k} = \lim_{k\to\infty} -\frac{1}{2}\frac{(d^k)^T\nabla^2 f(\mu^k)d^k}{(d^k)^T\nabla^2 f(x^k)d^k} = -\frac{1}{2}$$

Which gives

$$\lim_{k\to\infty}\frac{f(x^k+d^k)-f(x^k)}{\nabla f(x^k)^T d^k} = \frac{1}{2}$$

Since $\sigma \in (0, 1/2)$, that implies that for k large enough we have

$$\frac{f(x^k+d^k)-f(x^k)}{\nabla f(x^k)^T d^k} \geq \sigma$$

which gives (since $\nabla f(x^k)^T d^k < 0$)

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

That implies that eventually the full step-size is accepted (namely $t_k = 1$ for k large enough). Therefore the method eventually becomes the classic Newton's method. Since $\nabla^2 f$ is continuously differentiable on $\mathcal{L}(x^0)$, it is also locally Lipschitz on $\mathcal{L}(x^0)$. This implies that our algorithm is locally quadratically convergent. $\square$

Numerical Results for the Unconstrained Case

We now present numerical results for Algorithm 1. We use the same parameters as in the original paper [1], namely $\rho = 0.5$, $\sigma = 10^{-4}$ and $\epsilon = 10^{-5}$. We also used the nonmonotone line search as suggested. The matrices we work with are of the form

$$H := \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \theta \end{pmatrix} \in \mathbb{R}^{n\times n}$$

where $\theta > 0$. We are interested in the convergence of the algorithm when θ goes to 0 or infinity. We run the code for $m = 10$ and $m = 100$ anchor points. We do 100 test runs for each settings of the parameter. We consider that the algorithm fails when the number of iterations is larger than 1000.

The first observation is that we are able to obtain the same results as [1] when $\theta = 1$. When θ becomes small, the average number of iterations stays similar, between 2 and 3. When θ is larger, the average number of iterations goes up to around 5.

Table 1. Numerical results for $m = 10$

n		$\theta = 0.0001$	$\theta = 0.001$	$\theta = 0.01$	$\theta = 0.1$	$\theta = 1$
2	av. iter.	3.91	3.61	3.08	3.28	3.31
	nb of anchor solution	89	74	49	3	16
	nb of failures	0	0	0	0	0
	av. cpu time	0.0054	0.0021	0.0013	0.001	0.0011
3	av. iter.	3.38	3.28	3.27	3.27	3.37
	nb of anchor solution	13	15	11	6	3
	nb of failures	0	0	0	0	0
	av. cpu time	0.0010	0.00097	9.59e-4	9.35e-4	8.75e-4
4	av. iter.	3.35	3.41	3.31	3.39	3.55
	nb of anchor solution	5	3	2	0	1
	nb of failures	0	0	0	0	0
	av. cpu time	0.0011	0.0014	0.0017	0.0018	0.001
6	av. iter.	3.67	3.7	3.66	3.76	3.82
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	8.38e-4	0.0015	0.0021	0.0017	0.0015
8	av. iter.	3.89	3.94	3.85	3.87	4
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0017	0.0017	0.0015	0.0015	0.0013
10	av. iter.	3.97	3.98	3.98	4.03	4.01
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0013	0.0012	9.39e-4	0.001	0.001

Table 2. Numerical results for $m = 10$

n		$\theta = 10$	$\theta = 100$	$\theta = 1000$	$\theta = 10^4$
2	av. iter.	3.67	4.19	4.67	5.2
	nb of anchor solution	21	41	70	85
	nb of failures	0	0	0	0
	av. cpu time	0.0011	0.0011	0.0012	0.0018
3	av. iter.	3.70	4.14	4.37	5.07
	nb of anchor solution	8	21	59	85
	nb of failures	0	0	0	0
	av. cpu time	0.001	8.70e-4	0.0011	0.0016
4	av. iter.	3.79	4.07	4.46	5
	nb of anchor solution	3	11	43	87
	nb of failures	0	0	0	0
	av. cpu time	0.001	9.77e-4	0.0011	0.0016
6	av. iter.	3.96	4.21	4.46	5.04
	nb of anchor solution	0	5	33	75
	nb of failures	0	0	0	0
	av. cpu time	0.0014	0.0016	0.002	0.0025
8	av. iter.	4.01	4.30	4.55	5.13
	nb of anchor solution	0	1	29	68
	nb of failures	0	0	0	0
	av. cpu time	0.0013	0.0013	0.0014	0.0016
10	av. iter.	4.04	4.22	4.41	4.92
	nb of anchor solution	1	0	30	74
	nb of failures	0	0	0	1
	av. cpu time	8.94e-4	9.55e-4	0.001	0.0208

Table 3. Numerical results for $m = 100$

n		$\theta = 0.0001$	$\theta = 0.001$	$\theta = 0.01$	$\theta = 0.1$	$\theta = 1$
2	av. iter.	3.72	3.51	3.24	3.05	2.93
	nb of anchor solution	19	8	1	4	0
	nb of failures	2	0	0	0	0
	av. cpu time	0.2208	0.0072	0.0082	0.0064	0.0056
3	av. iter.	2.97	2.96	2.99	2.91	2.86
	nb of anchor solution	0	1	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0043	0.0083	0.0067	0.0081	0.0069
4	av. iter.	2.78	2.77	2.88	2.85	2.94
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0052	0.0049	0.0048	0.0046	0.0053
6	av. iter.	2.98	2.98	2.98	2.99	3
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.005	0.0082	0.0061	0.0064	0.0055
8	av. iter.	3.04	3.07	3.11	3.1	3.27
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0065	0.0065	0.0062	0.0056	0.0072
10	av. iter.	3.38	3.35	3.34	3.32	3.58
	nb of anchor solution	0	0	0	0	0
	nb of failures	0	0	0	0	0
	av. cpu time	0.0076	0.0075	0.0072	0.007	0.0076

Table 4. Numerical results for $m = 100$

n		$\theta = 10$	$\theta = 100$	$\theta = 1000$	$\theta = 10^4$
2	av. iter.	3.08	3.5	4.18	4.40
	nb of anchor solution	1	2	3	22
	nb of failures	1	0	0	0
	av. cpu time	0.0059	0.0114	0.0113	0.0067
3	av. iter.	3.06	3.62	4.09	4.70
	nb of anchor solution	0	0	1	7
	nb of failures	0	0	0	0
	av. cpu time	0.0099	0.0118	0.0102	0.0098
4	av. iter.	3.1	3.66	4.09	4.65
	nb of anchor solution	0	0	0	1
	nb of failures	0	0	0	0
	av. cpu time	0.0049	0.0062	0.0065	0.0073
6	av. iter.	3.1	3.7	4.09	4.65
	nb of anchor solution	0	0	0	0
	nb of failures	0	0	0	0
	av. cpu time	0.0059	0.0078	0.0084	0.0093
8	av. iter.	3.26	3.77	4.14	4.53
	nb of anchor solution	0	0	0	0
	nb of failures	0	0	0	0
	av. cpu time	0.0057	0.0075	0.0082	0.0094
10	av. iter.	3.6	3.88	4.19	4.68
	nb of anchor solution	0	0	0	0
	nb of failures	0	0	0	0
	av. cpu time	0.009	0.0077	0.0077	0.0085

4 Box Constraints

In this section, we are interested in the problem

$$\min f(x) := \sum \omega_i \left\|x - a^i\right\|_H \qquad \text{subject to } x \in X = \left\{x \in \mathbb{R}^n \mid b \le x \le c\right\} \tag{7}$$

where $b, c \in \mathbb{R}^n$ and inequalities are component-wise. We assume that for all $j = 1, \ldots, n$, we have $b_j \le c_j$, so that X is non-empty. We would like to apply the projected Newton method from [4] to the problem. Before we present the algorithm, we define the projection onto the set X as

$$P_X(x)_j = \begin{cases} b_j & \text{if } x_j < b_j \\ x_j & \text{if } b_j \le x_j \le c_j \\ c_j & \text{otherwise} \end{cases}$$

The algorithm is then as follows.

Algorithm 2 Projected Newton method for the box-constrained Fermat-Weber problem

(S0) Determine $p \in K$ such that $f(a^p) = \min_{i \in K} f(a^i)$. If a^p is such that

$$\left\| \sum_{i=1, i \neq p}^{m} \omega_i \frac{S(a^p - a^i)}{\|S(a^p - a^i)\|} \right\| \leq \omega_p$$

then a^p is the minimum. STOP.

(S1) Compute

$$d_p := \frac{-R_p}{\|R_p\|_H} \quad \text{with} \quad R_p := \sum_{i=1, i \neq p}^{m} \omega_i \frac{a^p - a^i}{\|a^p - a^i\|_H}$$

and find a step-size $t_p > 0$ such that $f(a^p + t_p d_p) < f(a_p)$ by backtracking. Set $x^0 := a^p + t_p d_p$, $k := 0$, and choose parameter $\epsilon > 0$, $\beta \in (0,1)$, $\sigma \in (0, 1/2)$ and a fixed positive definite matrix $M \in \mathbb{R}^{n \times n}$.

(S2) Denote

$$u_k = |x^k - P_X(x^k - M\nabla f(x^k))|$$

and set $\epsilon_k = \min\{u_k, \epsilon\}$. Define the set J_k as

$$J_k := \left\{ j \in \{1, \ldots, n\} \mid \begin{cases} b_j \leq x_j^k \leq b_j + \epsilon_k \text{ and } \frac{\partial f(x^k)}{\partial x_j} > 0 & \text{or} \\ c_j - \epsilon_k \leq x_j^k \leq c_j \text{ and } \frac{\partial f(x^k)}{\partial x_j} < 0 \end{cases} \right\}$$

Set $D^k \in \mathbb{R}^{n \times n}$ as $D^k = (H^k)^{-1}$ where H_k is given by

$$H_{ij}^k = \begin{cases} 0, & \text{if } i \neq j, \text{ and either } i \in J_k \text{ or } j \in J_k \\ \frac{\partial f(x^k)}{\partial x_i x_j}, & \text{otherwise.} \end{cases}$$

and denote

$$p^k = D_k \nabla f(x^k).$$

(S3) Define the following function

$$x^k(\alpha) = P_X(x^k - \alpha p^k) \quad (\alpha \geq 0)$$

Determine $m_k \in \mathbb{N}$ so that m_k is the first nonnegative integer m such that

$$f(x^k) - f(x^k(\beta^m)) \geq \sigma\left(\beta^m \sum_{j \notin J_k} \frac{\partial f(x^k)}{\partial x_j} p_j^k + \sum_{j \in J_k} \frac{\partial f(x^k)}{\partial x_j}(x_j^k - x_j^k(\beta^m))\right).$$

(S4) Set $x^{k+1} = x^k(\beta^{m_k})$, $k = k+1$, and go to (S2).

It is known from [4] that the projected Newton method decreases the value of the objective function at each iteration. Hence starting at a value $x^0 \in X$ such that $f(x^0) < f(a^i)$ for all $a^i \in X$ ensures us that the algorithm will never reach a point of non-differentiability (since each iterate of projected Newton algorithm remains in

X). Therefore our algorithm is well-defined and inherits the convergence properties of the projected Newton algorithm. We now explore these properties.

Proposition 4.1. *Let x^* be the solution of the Fermat Weber problem with box constraints (7) and assume $x^* \neq a^i$ for all $i \in \{1, \dots, m\}$. Moreover, assume that*

$$\frac{\partial f(x^*)}{\partial x_j} > 0 \text{ if } j \in \left\{ j \in \{1, \dots, n\} \quad | \quad x_j^* = b_j \right\}$$
$$\frac{\partial f(x^*)}{\partial x_j} < 0 \text{ if } j \in \left\{ j \in \{1, \dots, n\} \quad | \quad x_j^* = c_j \right\}$$

Then the sequence $\{x^k\}$ converges to x^* and the rate of convergence is quadratic.

Proof. This result follows from 3.2 and the fact that since f is smooth on a neighborhood of x^*, the Hessian $\nabla^2 f$ is Lipschitz continuous on in that neighborhood. We can then conclude with [4, Proposition 4]

□

Finally, observe that we did not specify a stopping criterion for Algorithm 2. This is because the literature does not give a specific criterion in general, and in the next subsection we explore the results with two different possible stopping criteria.

Numerical Results for Box Constraints

We would like to compare the projected Newton method to a projected accelerated gradient method, FISTA [8]. The FISTA method in the special case of the Fermat-Weber problem (7) goes as follows.

Algorithm 3 FISTA for box-constrained Fermat-Weber problem

```
(S0) Choose $x^0 \in \mathbb{R}^n$. Set $y^1 = x^0$, $t_1 = 1$ and choose $L$, a Lipschitz constant of
$\nabla f$. Set $k := 1$
(S1) Do the following updates:
        $$x^k = P_X(y^k - \frac{1}{L}\nabla f(y^k)),$$
        $$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{t_{k+1}},$$
        $$y^{k+1} = x^k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1})$$
(S2) Set $k = k + 1$, and go to (S1).
```

We compare both methods for $m = 10$ anchor points. We again work with matrices of the form

$$H := \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \theta \end{pmatrix} \in \mathbb{R}^{n \times n}$$

where $\theta > 0$. The Lipschitz constant used for FISTA is chosen to be

$$L = 2\|S\| \sum_{i=1}^{m} \omega_i$$

For each setting of the parameters, we generated 100 random instances of the Fermat Weber problem and ran both algorithms on each of the instances.
We consider that the algorithms fail when the number of iterations exceeds 50,000. We tested two different stopping criteria, $\left\|x^{k+1} - x^k\right\| < 10^{-8}$ and $\left|f(x^{k+1}) - f(x^k)\right| < 10^{-8}$.

Table 5. Numerical results for stopping criterion $\left\|x^{k+1} - x^k\right\| < 10^{-8}$

n		$\theta = 0.0001$	$\theta = 0.001$	$\theta = 0.01$	$\theta = 0.1$	$\theta = 1$
2	av. iter. Newton	4.57	4.36	3.70	3.59	3.44
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	4,706	7,008	3,283	2,000	1,013
	nb of failures fista	3	0	0	0	0
	nb of anchor solution	9	13	4	3	0
4	av. iter. Newton	4.05	4.31	4.00	4.13	3.93
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	10,563	11,185	6,009	3,080	1,948
	nb of failures fista	0	1	0	0	0
	nb of anchor solution	0	0	0	0	0
6	av. iter. Newton	4.1	4.03	4.17	4.04	4.04
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	13,154	10,397	7,024	4,207	2,755
	nb of failures fista	5	0	0	0	0
	nb of anchor solution	0	0	0	0	0
8	av. iter. Newton	3.99	4.25	4.02	4.01	4.1
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	10,287	10,767	9,363	4,536	3,437
	nb of failures fista	1	1	0	0	0
	nb of anchor solution	0	0	0	0	0
10	av. iter. Newton	3.96	4.07	3.97	4.03	3.94
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	12,753	13,172	10,080	5,407	3,997
	nb of failures fista	5	0	0	0	0
	nb of anchor solution	0	0	0	0	0

Table 6. Numerical results for stopping criterion $\left\|x^{k+1} - x^k\right\| < 10^{-8}$

n		$\theta = 10$	$\theta = 100$	$\theta = 1000$	$\theta = 10^4$
2	av. iter. Newton	3.83	4.27	4.03	4.40
	nb of failures Newton	0	0	0	0
	av. iter. fista	1,813	3,206	6,098	6,301
	nb of failures fista	1	1	1	5
	nb of anchor solution	2	11	11	9
4	av. iter. Newton	4.05	4.47	5.06	5.89
	nb of failures Newton	0	0	0	0
	av. iter. fista	5,074	15,733	11,509	15,499
	nb of failures fista	0	1	34	39
	nb of anchor solution	0	0	1	1
6	av. iter. Newton	4.15	4.81	5.3	5.97
	nb of failures Newton	0	0	0	0
	av. iter. fista	8,327	27,167	11,927	21,808
	nb of failures fista	0	7	69	79
	nb of anchor solution	0	0	0	16
8	av. iter. Newton	4.18	4.62	5.26	5.72
	nb of failures Newton	0	0	0	0
	av. iter. fista	10,927	32,762	20,694	20,372
	nb of failures fista	0	25	88	88
	nb of anchor solution	0	0	0	0
10	av. iter. Newton	4.01	4.44	5.11	5.21
	nb of failures Newton	0	0	0	0
	av. iter. fista	12,282	39,561	15,968	8,819
	nb of failures fista	0	49	93	97
	nb of anchor solution	0	0	0	0

Table 7. Numerical results for stopping criterion $\left|f(x^{k+1}) - f(x^k)\right| < 10^{-8}$

n		$\theta = 0.0001$	$\theta = 0.001$	$\theta = 0.01$	$\theta = 0.1$	$\theta = 1$
2	av. iter. Newton	3.5	3.59	3.54	2.97	3.03
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	1,317	994	862	540	448
	nb of failures fista	2	2	1	0	1
	nb of anchor solution	14	13	7	7	2
4	av. iter. Newton	3.55	3.49	3.33	3.48	3.40
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	2,346	1,497	1,414	1,038	747
	nb of failures fista	0	0	0	0	0
	nb of anchor solution	0	0	0	0	0
6	av. iter. Newton	3.28	3.27	3.36	3.33	3.21
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	2,586	1,530	1,266	1,256	852
	nb of failures fista	0	0	0	0	0
	nb of anchor solution	0	0	0	0	0
8	av. iter. Newton	3.2	3.21	3.29	3.31	3.23
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	2,895	1,478	1,433	1,274	968
	nb of failures fista	0	0	0	0	0
	nb of anchor solution	0	0	0	0	0
10	av. iter. Newton	3.21	3.31	3.31	3.29	3.25
	nb of failures Newton	0	0	0	0	0
	av. iter. fista	3,030	1,746	1,603	1,481	1,117
	nb of failures fista	0	0	0	0	0
	nb of anchor solution	0	0	0	0	0

TABLE 8. Numerical results for stopping criterion $\left|f(x^{k+1}) - f(x^k)\right| < 10^{-8}$

n		$\theta = 10$	$\theta = 100$	$\theta = 1000$	$\theta = 10^4$
2	av. iter. Newton	3.15	3.65	3.57	3.59
	nb of failures Newton	0	0	0	0
	av. iter. fista	901	1,380	2,372	2,436
	nb of failures fista	1	0	1	0
	nb of anchor solution	1	6	12	19
4	av. iter. Newton	3.62	3.84	4.47	4.62
	nb of failures Newton	0	0	0	0
	av. iter. fista	1,524	2,955	5,131	8,355
	nb of failures fista	0	0	0	0
	nb of anchor solution	0	1	0	1
6	av. iter. Newton	3.41	3.95	4.33	4.64
	nb of failures Newton	0	0	0	0
	av. iter. fista	1,920	4,248	6,811	8,983
	nb of failures fista	0	0	0	0
	nb of anchor solution	0	0	0	0
8	av. iter. Newton	3.41	3.78	4.45	4.83
	nb of failures Newton	0	0	0	0
	av. iter. fista	2,159	4,442	7,228	10,796
	nb of failures fista	0	0	0	0
	nb of anchor solution	0	0	0	0
10	av. iter. Newton	3.24	3.86	4.57	4.98
	nb of failures Newton	0	0	0	0
	av. iter. fista	2,318	4,707	8,480	12,088
	nb of failures fista	0	0	0	0
	nb of anchor solution	0	0	0	0

The results show that the projected Newton method beats a classic algorithm like FISTA by several orders of magnitude. However, it is interesting to note that FISTA seems to approach the optimal solution a lot faster than what the numerical results show. Figure 1 shows that about 100 iterations are needed to reach a satisfying solution.

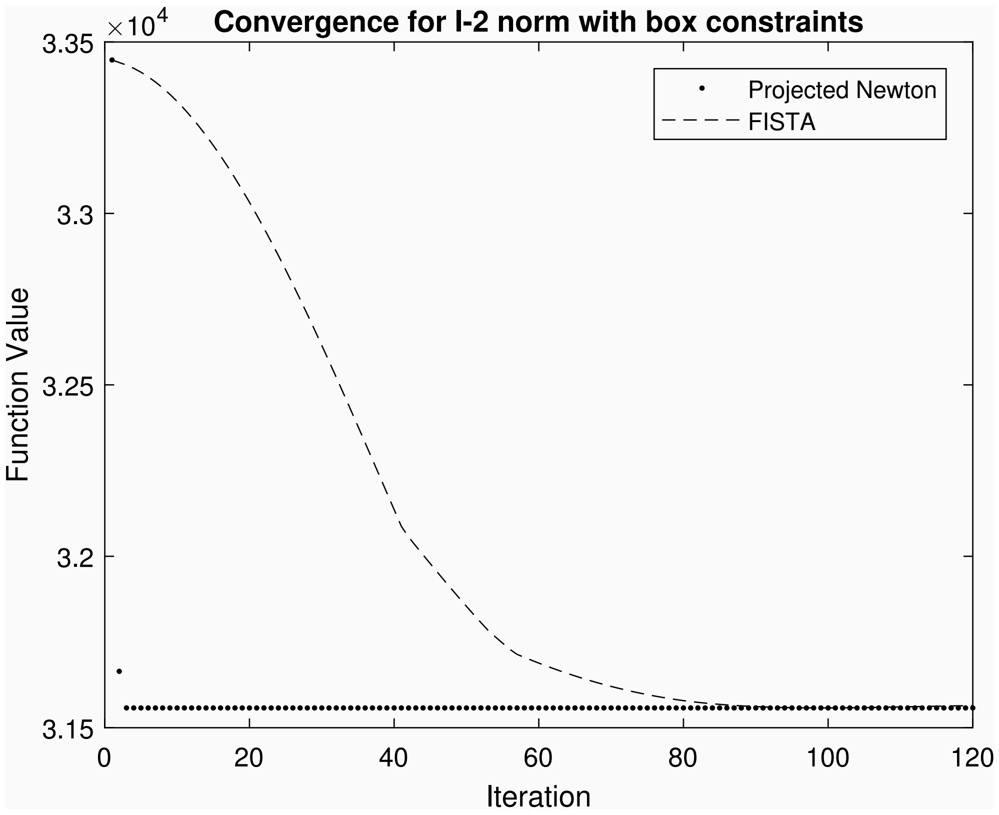

FIGURE 1. Typical example for $n = 8$, $m = 10$ and $\theta = 10$

References

[1] S. Görner and C. Kanzow: *On Newtons Method for the FermatWeber Location Problem.* Journal of Optimization Theory and Applications 170, 107-118 (2016).

[2] A. Beck and S. Sabach *Weiszfelds method: old and new results.* Journal of Optimization Theory and Applications 164, 140 (2015)

[3] J. Nocedal and S.J. Wright:*Numerical Optimization*, 2nd edn. Springer, New York (2006)

[4] D. P. Bertsekas: *Projected Newton Methods for Optimization Problems with Simple Constraints*, SIAM Journal on Control and Optimization, 20(2):221246, 1982

[5] M. Schmidt, D. Kim and S. Sra: *Projected Newton-type methods in machine learning*, Optimization for Machine Learning, MIT Press (2011).

[6] A. Beck,*First-Order Methods in Optimization*, Society for Industrial and Applied Mathematics (2017).

[7] S. Adrian, S. Lewis and M.L. Overton, *Nonsmooth optimization via quasi-newton methods*, Mathematical Programming, 141(1-2):135163 (2013).

[8] A. Beck and M. Teboulle *A fast iterative shrinkagethresholding algorithm with application to wavelet-based image deblurring*, in IEEE Int. Conf. Acoust., Speech, Signal Process, pp. 693696 (2009).

[9] J. B. Hiriart-Urruty and C. Lemarechal: Fundamentals of Convex Analysis, Springer, Berlin (2001)

[10] W. Warth and J. Werner: Effiziente Schrittweitenfunktionen bei unrestringierten Optimierungsaufgaben. Computing 19, 5972 (1977)

McGill University, Department of Mathematics, Student ID : 260673170
E-mail address: `benjamin.paul-dubois-taine@mail.mcgill.ca`