Geometric Median in Nearly Linear Time

Michael B. Cohen
Massachusetts Institute of Technology
USA
micohen@mit.edu

Yin Tat Lee
Massachusetts Institute of Technology
USA
yintat@mit.edu

Gary Miller
Carnegie Mellon University
USA
glmiller@cs.cmu.edu

Jakub Pachocki
Carnegie Mellon University
USA
pachocki@cs.cmu.edu

Aaron Sidford
Microsoft Research New England
USA
asid@microsoft.com

ABSTRACT

In this paper we provide faster algorithms for solving the geometric median problem: given n points in $\mathbb{R}^d$ compute a point that minimizes the sum of Euclidean distances to the points. This is one of the oldest non-trivial problems in computational geometry yet despite a long history of research the previous fastest running times for computing a $(1+\epsilon)$-approximate geometric median were $O(d \cdot n^{4/3}\epsilon^{-8/3})$ by Chin et. al, $\tilde{O}(d \exp \epsilon^{-4} \log \epsilon^{-1})$ by Badoiu et. al, $O(nd + \text{poly}(d, \epsilon^{-1}))$ by Feldman and Langberg, and the polynomial running time of $O((nd)^{O(1)} \log \frac{1}{\epsilon})$ by Parrilo and Sturmfels and Xue and Ye.

In this paper we show how to compute such an approximate geometric median in time $O(nd \log^3 \frac{n}{\epsilon})$ and $O(d\epsilon^{-2})$. While our $O(d\epsilon^{-2})$ is a fairly straightforward application of stochastic subgradient descent, our $O(nd \log^3 \frac{n}{\epsilon})$ time algorithm is a novel long step interior point method. We start with a simple $O((nd)^{O(1)} \log \frac{1}{\epsilon})$ time interior point method and show how to improve it, ultimately building an algorithm that is quite non-standard from the perspective of interior point literature. Our result is one of few cases of outperforming standard interior point theory. Furthermore, it is the only case we know of where interior point methods yield a nearly linear time algorithm for a canonical optimization problem that traditionally requires superlinear time.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*Convex programming*; F.2.0 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Algorithms, Theory

STOC'16, June 19–21, 2016, Cambridge, MA, USA

http://dx.doi.org/10.1145/2897518.2897647

Keywords

geometric median, interior point methods, stochastic gradient descent

1. INTRODUCTION

One of the oldest easily-stated nontrivial problems in computational geometry is the Fermat-Weber problem: given a set of n points in d *dimensions*, $a^{(1)}, \ldots, a^{(n)} \in \mathbb{R}^d$, find a point $x^* \in \mathbb{R}^d$ that minimizes the sum of Euclidean distances to them:

$$x^* \in \arg\min_{x \in \mathbb{R}^d} f(x) \quad \text{where} \quad f(x) \stackrel{\text{def}}{=} \sum_{i \in [n]} \|x - a^{(i)}\|_2$$

This problem, also known as the *geometric median problem,* is well studied and has numerous applications. It is often considered over low dimensional spaces in the context of the facility location problem [29] and over higher dimensional spaces it has applications to clustering in machine learning and data analysis. For example, computing the geometric median is a subroutine in popular expectation maximization heuristics for k-medians clustering.

The problem is also important to robust estimation, where we like to find a point representative of given set of points that is resistant to outliers. The geometric median is a rotation and translation invariant estimator that achieves the optimal *breakdown point* of 0.5, i.e. it is a good estimator even when up to half of the input data is arbitrarily corrupted [18]. Moreover, if a large constant fraction of the points lie in a ball of diameter ϵ then the geometric median lies in that ball with diameter $O(\epsilon)$ (see Lemma A.1). Consequently, the geometric median can be used to turn expected results into high probability results: e.g. if the $a^{(i)}$ are drawn independently such that $\mathbb{E}\|x - x^*\|_2 \leq \epsilon$ for some $\epsilon > 0$ and $x \in \mathbb{R}^d$ then this fact, Markov bound, and Chernoff Bound, imply $\|x^* - a^{(i)}\|_2 = O(\epsilon)$ with high probability in n.

Despite the ancient nature of the Fermat-Weber problem and its many uses there are relatively few theoretical guarantees for solving it (see Table 1). To compute a $(1+\epsilon)$-approximate solution, i.e. $x \in \mathbb{R}^d$ with $f(x) \leq (1+\epsilon)f(x^*)$, the previous fastest running times were either $O(d \cdot n^{4/3}\epsilon^{-8/3})$ by [7], $\tilde{O}(d \exp \epsilon^{-4} \log \epsilon^{-1})$ by [1], $\tilde{O}(nd + \text{poly}(d, \epsilon^{-1}))$ by [10], or $O((nd)^{O(1)} \log \frac{1}{\epsilon})$ time by [24, 31]. In this paper we improve upon these running times by provid-

ing an $O(nd \log^3 \frac{n}{\epsilon})$ time algorithm[1] as well as an $O(d/\epsilon^2)$ time algorithm, provided we have an oracle for sampling a random $a^{(i)}$. Picking the faster algorithm for the particular value of ϵ improves the running time to $O(nd \log^3 \frac{1}{\epsilon})$. We also extend these results to compute a $(1+\epsilon)$-approximate solution to the more general Weber's problem,

$$\min_{x \in \mathbb{R}^d} \sum_{i \in [n]} w_i \|x - a^{(i)}\|_2$$

for non-negative w_i, in time $O(nd \log^3 \frac{1}{\epsilon})$ (see Appendix D).

Our $O(nd \log^3 \frac{n}{\epsilon})$ time algorithm is a careful modification of standard interior point methods for solving the geometric median problem. We provide a long step interior point method tailored to the geometric median problem for which we implement every iteration in nearly linear time. While our analysis starts with a simple $O((nd)^{O(1)} \log \frac{1}{\epsilon})$ time interior point method and shows how to improve it, our final algorithm is quite non-standard from the perspective of interior point literature. Our result is one of very few cases we are aware of outperforming traditional interior point theory [20, 17] and the only we are aware of using interior point methods to obtain a nearly linear time algorithm for a canonical optimization problem that traditionally requires superlinear time. We hope our work leads to further improvements in this line of research.

Our $O(d\epsilon^{-2})$ algorithm is a relatively straightforward application of sampling techniques and stochastic subgradient descent. Some additional insight is required simply to provide a rigorous analysis of the robustness of the geometric median and use this to streamline our application of stochastic subgradient descent. We include it for completeness however, we defer its proof to Appendix A. The bulk of the work in this paper is focused on developing our $O(nd \log^3 \frac{n}{\epsilon})$ time algorithm which we believe uses a set of techniques of independent interest.

1.1 Previous Work

The geometric median problem was first formulated for the case of three points in the early 1600s by Pierre de Fermat [14, 9]. A simple elegant ruler and compass construction was given in the same century by Evangelista Torricelli. Such a construction does not generalize when a larger number of points is considered: Bajaj has shown the even for five points, the geometric median is not expressible by radicals over the rationals [2]. Hence, the $(1+\epsilon)$-approximate problem has been studied for larger values of n.

Many authors have proposed algorithms that achieve running times polynomial in n, d and $1/\epsilon$. The most cited and used algorithm is Weiszfeld's 1937 algorithm [30]. Unfortunately Weiszfeld's algorithm may not converge and if it does it may do so very slowly. There have been many proposed modifications to Weiszfeld's algorithm [8, 25, 23, 3, 27, 16] that in general give non-asymptotic runtime guarantees. In light of more modern multiplicative weights methods his algorithm can be viewed as a re-weighted least squares Chin et al. [7] considered the more general L_2 embedding problem: placing the vertices of a graph into $\mathbb{R}^d$, where some of the vertices have fixed positions while the remaining vertices are allowed to float, with the objective of minimizing the sum of the Euclidean edge lengths. Using the multiplicative weights method, they obtained a run time of $O(d \cdot n^{4/3}\epsilon^{-8/3})$ for a broad class of problems, including the geometric median problem.[2]

Many authors consider problems that are generalizations of the Fermat-Weber problem, and obtain algorithms for finding the geometric median as a specialization. For example, Badoiu et al. give an approximate k-median algorithm by sub-sampling with the runtime for $k = 1$ of $\widetilde{O}(d \cdot \exp(O(\epsilon^{-4})))$ [1]. Parrilo and Sturmfels demonstrated that the problem can be reduced to semidefinite programming, thus obtaining a runtime of $\widetilde{O}(\text{poly}(n,d)\log \epsilon^{-1})$ [24]. Furthermore, Bose et al. gave a linear time algorithm for fixed d and ϵ^{-1}, based on low-dimensional data structures [4] and it has been show how to obtain running times of $\widetilde{O}(nd + \text{poly}(d, \epsilon^{-1}))$ for this problem and a more general class of problems.[12, 10].

An approach very related to ours was studied by Xue and Ye [31]. They give an interior point method with barrier analysis that runs in time $\tilde{O}((d^3 + d^2 n)\sqrt{n} \log \epsilon^{-1})$.

1.2 Overview of $O(nd \log^3 \frac{n}{\epsilon})$ Time Algorithm

1.2.1 Interior Point Primer

Our algorithm is broadly inspired by interior point methods, a broad class of methods for efficiently solving convex optimization problems [32, 22]. Given an instance of the geometric median problem we first put the problem in a more natural form for applying interior point methods. Rather than writing the problem as minimizing a convex function over $\mathbb{R}^d$

$$\min_{x \in \mathbb{R}^d} f(x) \quad \text{where} \quad f(x) \stackrel{\text{def}}{=} \sum_{i \in [n]} \|x - a^{(i)}\|_2 \tag{1}$$

we instead write the problem as minimizing a linear function over a larger convex space:

$$\min_{\{\alpha, x\} \in S} 1^\top \alpha. \tag{2}$$

where

$$S \stackrel{\text{def}}{=} \left\{ \alpha \in \mathbb{R}^n, x \in \mathbb{R}^d \;\mid\; \|x^{(i)} - a^{(i)}\|_2 \le \alpha_i \text{ for all } i \in [n] \right\}$$

Clearly, these problems are the same as at optimality $\alpha_i = \|x^{(i)} - a^{(i)}\|_2$.

To solve problems of the form (2) interior point methods relax the constraint $\{\alpha, x\} \in S$ through the introduction of a *barrier function*. In particular they assume that there is a real valued function p such that as $\{\alpha, x\}$ moves towards the boundary of S the value of p goes to infinity. A popular class of interior point methods known as *path following methods [26, 11]*, they consider relaxations of (2) of the form

$$\min_{\{\alpha, x\} \in \mathbb{R}^n \times \mathbb{R}^d} t \cdot 1^\top \alpha + p(\alpha, x) \ .$$

The minimizers of this function form a path, known as the central path, parameterized by t. The methods then use variants of Newton's method to follow the path until t is large enough that a high quality approximate solution is obtained. The number of iterations of these methods are then

[1] If z is the total number of nonzero entries in the coordinates of the $a^{(i)}$ then a careful analysis of our algorithm improves our running time to $O(z \log^3 \frac{n}{\epsilon})$.

[2] The result of [7] was stated in more general terms than given here. However, it easy to formulate the geometric median problem in their model.

Year	Authors	Runtime	Comments
1659	Torricelli [28]	-	Assuming $n = 3$
1937	Weiszfeld [30]	-	Does not always converge
1990	Chandrasekaran and Tamir[6]	$\widetilde{O}(n \cdot \text{poly}(d) \log \epsilon^{-1})$	Ellipsoid method
1997	Xue and Ye [31]	$\widetilde{O}((d^3 + d^2 n)\sqrt{n} \log \epsilon^{-1})$	Interior point with barrier method
2000	Indyk [13]	$\widetilde{O}(dn \cdot \epsilon^{-2})$	Optimizes only over x in the input
2001	Parrilo and Sturmfels [24]	$\widetilde{O}(\text{poly}(n, d) \log \epsilon^{-1})$	Reduction to SDP
2002	Badoiu et al. [1]	$\widetilde{O}(d \cdot \exp(O(\epsilon^{-4})))$	Sampling
2003	Bose et al. [4]	$\widetilde{O}(n)$	Assuming $d, \epsilon^{-1} = O(1)$
2005	Har-Peled and Kushal [12]	$\widetilde{O}(n + \text{poly}(\epsilon^{-1}))$	Assuming $d = O(1)$
2011	Feldman and Langberg [10]	$\widetilde{O}(nd + \text{poly}(d, \epsilon^{-1}))$	Coreset
2013	Chin et al. [7]	$\widetilde{O}(dn^{4/3} \cdot \epsilon^{-8/3})$	Multiplicative weights
-	**This paper**	$O(nd \log^3(n/\epsilon))$	Interior point with custom analysis
-	**This paper**	$O(d\epsilon^{-2})$	Stochastic gradient descent

Table 1: Selected Previous Results.

typically governed by a property of p known as its self concordance ν. Given a ν-self concordant barrier, typically interior point methods require $O(\sqrt{\nu} \log \frac{1}{\epsilon})$ iterations to compute a $(1+\epsilon)$-approximate solution.

For our particular convex set, the construction of our barrier function is particularly simple, we consider each constraint $\|x - a^{(i)}\|_2 \leq \alpha_i$ individually. In particular, it is known that the function

$$p^{(i)}(\alpha, x) = -\ln \alpha_i - \ln\left(\alpha_i^2 - \|x - a^{(i)}\|_2\right)$$

is a $O(1)$ self-concordant barrier function for the set

$$S^{(i)} = \left\{x \in \mathbb{R}^d, \alpha \in \mathbb{R}^n \mid \|x - a^{(i)}\|_2 \leq \alpha_i\right\} \text{ [21].}$$

Since $\cap_{i\in[n]} S^{(i)} = S$ we can use the barrier $\sum_{i\in[n]} p^{(i)}(\alpha, x)$ for $p(\alpha, x)$ and standard self-concordance theory shows that this is an $O(n)$ self concordant barrier for S. Consequently, this easily yields an interior point method for solving the geometric median problem in $O((nd)^{O(1)} \log \frac{1}{\epsilon})$ time.

1.2.2 Difficulties

Unfortunately obtaining a nearly linear time algorithm for geometric median using interior point methods as presented poses numerous difficulties. Particularly troubling is the number of iterations required by standard interior point algorithms. The approach outlined in the previous section produced an $O(n)$-self concordant barrier and even if we use more advanced self concordance machinery, i.e. the universal barrier [22], the best known self concordance of barrier for the convex set $\sum_{i\in[n]} \|x - a^{(i)}\|_2 \leq c$ is $O(d)$. An interesting open question still left open by our work is to determine what is the minimal self concordance of a barrier for this set.

Consequently, even if we could implement every iteration of an interior point scheme in nearly linear time it is unclear whether one should hope for a nearly linear time interior point algorithm for the geometric median. While there are a instances of outperforming standard self-concordance analysis [20, 17], these instances are few, complex, and to varying degrees specialized to the problems they solve. Moreover, we are unaware of any interior point scheme providing a provable nearly linear time for a general nontrivial convex optimization problem.

1.2.3 Beyond Standard Interior Point

Despite these difficulties we do obtain a nearly linear time interior point based algorithm that only requires $O(\log \frac{n}{\epsilon})$ iterations, i.e. increases to the path parameter. After choosing the natural penalty functions $p^{(i)}$ described above, we optimize in closed form over the α_i to obtain the following penalized objective function:[3]

$$\min_x f_t(x)$$

where $f_t(x)$ is defined to be

$$\sum_{i\in[n]} \sqrt{1 + t^2\|x - a^{(i)}\|_2^2} - \ln\left[1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}\right]$$

We then approximately minimize $f_t(x)$ for increasing t. We let

$$x_t \stackrel{\text{def}}{=} \arg\min_{x\in\mathbb{R}^d} f_t(x)$$

for $t \geq 0$, and thinking of $\{x_t : t \geq 0\}$ as a continuous curve known as the central path, we show how to approximately follow this path. As $\lim_{t\to\infty} x_t = x_*$ this approach yields a $(1+\epsilon)$-approximation.

So far our analysis is standard and interior point theory yields an $\Omega(\sqrt{n})$ iteration interior point scheme. To overcome this we take a more detailed look at x_t. We note that for any t if there is any rapid change in x_t it must occur in the direction of the smallest eigenvector of $\nabla^2 f_t(x)$, denoted v_t, what we henceforth may refer to as the *bad direction* at x_t. Furthermore, we show that this bad direction is *stable* in the sense that for all directions $d \perp v_t$ it is the case that $d^\top(x_t - x_{t'})$ is small for $t' \leq ct$ for a small constant c.

In fact, we show that this movement over such a *long step,* i.e. constant increase in t, in the directions orthogonal to the bad direction is small enough that for any movement around a ball of this size the Hessian of f_t only changes by a small multiplicative constant. In short, starting at x_t there exists a point y obtained just by moving from x_t in the bad direction, such that y is close enough to $x_{t'}$ that standard first order method will converge quickly to $x_{t'}$! Thus, we

[3]It is unclear how to extend our proof for the simpler function: $\sum_{i\in[n]} \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}$.

might hope to find such a y, quickly converge to $x_{t'}$ and repeat. If we increase t by a multiplicative constant in every such iterations, standard interior point theory suggests that $O(\log \frac{n}{\epsilon})$ iterations suffices.

1.2.4 Building an Algorithm

To turn the structural result in the previous section into a fast algorithm there are several further issues we need to address. We need to

- (1) Show how to find the point along the bad direction that is close to $x_{t'}$
- (2) Show how to solve linear systems in the Hessian to actually converge quickly to $x_{t'}$
- (3) Show how to find the bad direction
- (4) Bound the accuracy required by these computations

Deferring (1) for the moment, our solution to the rest are relatively straightforward. Careful inspection of the Hessian of f_t reveals that it is well approximated by a multiple of the identity matrix minus a rank 1 matrix. Consequently using explicit formulas for the inverse of of matrix under rank 1 updates, i.e. the Sherman-Morrison formula, we can solve such systems in nearly linear time thereby addressing (2). For (3), we show that the well known power method carefully applied to the Hessian yields the bad direction if it exists. Finally, for (4) we show that a constant approximate geometric median is near enough to the central path for $t = \Theta(\frac{1}{f(x_*)})$ and that it suffices to compute a central path point at $t = O(\frac{n}{f(x_*)\epsilon})$ to compute a $1+\epsilon$-geometric median. Moreover, for these values of t, the precision needed in other operations is clear.

The more difficult operation is (1). Given x_t and the bad direction exactly, it is still not clear how to find the point along the bad direction line from x_t that is close to $x_{t'}$. Just performing binary search on the objective function a priori might not yield such a point due to discrepancies between a ball in Euclidean norm and a ball in hessian norm and the size of the distance from the optimal point in euclidean norm. To overcome this issue we still line search on the bad direction, however rather than simply using $f(x_t + \alpha \cdot v_t)$ as the objective function to line search on, we use the function

$$g(\alpha) = \min_{\|x - x_t - \alpha \cdot v_t\|_2 \leq c} f(x)$$

for some constant c, that is given an α we move α in the bad direction and take the best objective function value in a ball around that point. For appropriate choice of c the minimizers of α will include the optimal point we are looking for. Moreover, we can show that g is convex and that it suffices to perform the minimization approximately.

Putting these pieces together yields our result. We perform $O(\log \frac{n}{\epsilon})$ iterations of interior point (i.e. increasing t), where in each iteration we spend $O(nd \log \frac{n}{\epsilon})$ time to compute a high quality approximation to the bad direction, and then we perform $O(\log \frac{n}{\epsilon})$ approximate evaluations on $g(\alpha)$ to binary search on the bad direction line, and then to approximately evaluate g we perform gradient descent in approximate Hessian norm to high precision which again takes $O(nd \log \frac{n}{\epsilon})$ time. Altogether this yields a $O(nd \log^3 \frac{n}{\epsilon})$ time algorithm to compute a $1 + \epsilon$ geometric median. Here we made minimal effort to improve the log factors and plan to investigate this further in future work.

1.3 Overview of $O(d\epsilon^{-2})$ Time Algorithm

In addition to providing a nearly linear time algorithm we provide a stand alone result on quickly computing a crude $(1+\epsilon)$-approximate geometric median in Section A. In particular, given an oracle for sampling a random $a^{(i)}$ we provide an $O(d\epsilon^{-2})$, i.e. sublinear, time algorithm that computes such an approximate median. Our algorithm for this result is fairly straightforward. First, we show that random sampling can be used to obtain some constant approximate information about the optimal point in constant time. In particular we show how this can be used to deduce an Euclidean ball which contains the optimal point. Second, we perform stochastic subgradient descent within this ball to achieve our desired result.

1.4 Paper Organization

The rest of the paper is structured as follows. After covering preliminaries in Section 2, in Section 3 we provide various results about the central path that we use to derive our nearly linear time algorithm. In Section 4 we then provide our nearly linear time algorithm. Many proofs and supporting lemmas for these sections are deferred to the appendix of the full version on arXiv. In Appendix A we provide our $O(d/\epsilon^2)$ algorithm, in Appendix B we provide the derivation of our penalized objective function, in Appendix C we provide general technical machinery we use throughout, and in Appendix D we show how to extend our results to the Weber problem, i.e. weighted geometric median.

2. NOTATION

2.1 General Notation

We use bold to denote a matrix. For a symmetric positive semidefinite matrix (PSD), $\mathbf{A}$, we let $\lambda_1(\mathbf{A}) \geq ... \geq \lambda_n(\mathbf{A}) \geq 0$ denote the eigenvalues of $\mathbf{A}$ and let $v_1(\mathbf{A}), ..., v_n(\mathbf{A})$ denote corresponding eigenvectors. We let $\|x\|_{\mathbf{A}} \stackrel{\text{def}}{=} \sqrt{x^\top \mathbf{A} x}$ and for PSD we use $\mathbf{A} \preceq \mathbf{B}$ and $\mathbf{B} \preceq \mathbf{A}$ to denote the conditions that $x^\top \mathbf{A} x \leq x^\top \mathbf{B} x$ for all x and $x^\top \mathbf{B} x \leq x^\top \mathbf{A} x$ for all x respectively.

2.2 Problem Notation

The central problem of this is as follows: we are given points $a^{(1)}, ..., a^{(n)} \in \mathbb{R}^d$ and we wish to compute a geometric median, i.e.

$$x_* \in \arg\min_x f(x)$$

where

$$f(x) = \sum_{i \in [n]} \|a^{(i)} - x\|_2 \ .$$

We call a point $x \in \mathbb{R}^d$ an $(1+\epsilon)$-approximate geometric median if $f(x) \leq (1+\epsilon) f(x_*)$.

2.3 Penalized Objective Notation

To solve this problem we relax the objective function f and instead consider the following family of penalized objective function parameterized by $t > 0$

$$\min_x f_t(x)$$

where $f_t(x)$ is given by

$$\sum_{i\in[n]} \sqrt{1+t^2\|x-a^{(i)}\|_2^2} - \ln\left[1+\sqrt{1+t^2\|x-a^{(i)}\|_2^2}\right]$$

This penalized objective function arrises naturally in considering a standard interior point formulation of the geometric median problem. (See Section B for the derivation.) For all *path parameters* $t > 0$, we let

$$x_t \stackrel{\text{def}}{=} \arg\min_x f_t(x) \ .$$

Our primary goal is to obtain good approximations to the *central path* $\{x_t : t > 0\}$ for increasing values of t.

We let

$$g_t^{(i)}(x) \stackrel{\text{def}}{=} \sqrt{1+t^2\|x-a^{(i)}\|_2^2}$$

and

$$f_t^{(i)}(x) \stackrel{\text{def}}{=} g_t^{(i)}(x) - \ln(1+g_t^{(i)}(x))$$

so

$$f_t(x) = \sum_{i\in[n]} f_t^{(i)}(x) \ .$$

We refer to the quantity

$$w_t(x) \stackrel{\text{def}}{=} \sum_{i\in[n]} \frac{1}{1+g_t^{(i)}(x)}$$

as weight as it is a natural measure of total contribution of the $a^{(i)}$ to $\nabla^2 f_t(x)$. We let

$$\bar{g}_t(x) \stackrel{\text{def}}{=} w_t(x)\left[\sum_{i\in[n]} \frac{1}{(1+g_t^{(i)}(x_t))g_t^{(i)}(x_t)}\right]^{-1}$$

denote a natural term that helps upper bound the rate of change of the central path. Furthermore we use

$$u^{(i)}(x) \stackrel{\text{def}}{=} \frac{x-a^{(i)}}{\|x-a^{(i)}\|_2}$$

to denote the unit vector corresponding to $x - a^{(i)}$. Finally we let $\mu_t(x) \stackrel{\text{def}}{=} \lambda_d(\nabla^2 f_t(x))$, the minimum eigenvalue of $\nabla^2 f_t(x)$, and let $v_t(x)$ denote a corresponding eigenvector. To simplify notation we often drop the (x) in these definitions when $x = x_t$ and t is clear from context.

3. PROPERTIES OF THE CENTRAL PATH

Here provide various facts regarding the penalized objective function and the central path. While we use the lemmas in this section throughout the paper, the main contribution of this section is Lemma 3.5 in Section 3.3. There we prove that with the exception of a single direction, the change in the central path is small over a constant multiplicative change in the path parameter. In addition, we show that our penalized objective function is stable under changes in a $O(\frac{1}{t})$ Euclidean ball (Section 3.1), and we bound the change in the Hessian over the central path (Section 3.2). Furthermore, in Section 3.4, we relate $f(x_t)$ to $f(x^*)$.

3.1 How Much Does the Hessian Change in General?

Here, we show that the Hessian of the penalized objective function is stable under changes in a $O(\frac{1}{t})$ sized Euclidean ball. This shows that if we have a point which is close to a central path point in Euclidean norm, then we can use Newton method to find it.

LEMMA 3.1. *Suppose that* $\|x-y\|_2 \le \frac{\epsilon}{t}$ *with* $\epsilon \le \frac{1}{20}$. *Then, we have*

$$(1-6\epsilon^{2/3})\nabla^2 f_t(x) \preceq \nabla^2 f_t(y) \preceq (1+6\epsilon^{2/3})\nabla^2 f_t(x).$$

3.2 How Much Does the Hessian Change on the Path?

Here we bound how much the Hessian of the penalized objective function can change along the central path. First we provide the following lemma bound several aspects of the penalized objective function and proving that the weight, w_t, only changes by a small amount multiplicatively given small multiplicative changes in the path parameter, t.

LEMMA 3.2. *For all* $t \ge 0$ *and* $i \in [n]$ *the following hold*

$$\left\|\frac{d}{dt}x_t\right\|_2 \le \frac{1}{t^2}\bar{g}_t(x_t) \quad , \quad \left|\frac{d}{dt}g_t^{(i)}(x_t)\right| \le \frac{1}{t}\left(g_t^{(i)}(x_t)+\bar{g}_t\right) \ ,$$

and

$$\left|\frac{d}{dt}w_t\right| \le \frac{2}{t}w_t$$

Consequently, for all $t' \ge t$ *we have that*

$$\left(\frac{t}{t'}\right)^2 w_t \le w_{t'} \le \left(\frac{t'}{t}\right)^2 w_t \ .$$

Next we use this lemma to bound the change in the Hessian with respect to t.

LEMMA 3.3. *For all* $t \ge 0$ *we have*

$$-12\cdot t\cdot w_t\mathbf{I} \preceq \frac{d}{dt}\left[\nabla^2 f_t(x_t)\right] \preceq 12\cdot t\cdot w_t\mathbf{I}$$

and therefore for all $\beta \in [0, \frac{1}{8}]$ *and* $t' = t(1+\beta)$

$$\nabla^2 f(x_t) - 15\beta t^2 w_t\mathbf{I} \preceq \nabla^2 f(x_{t'}) \preceq \nabla^2 f(x_t) + 15\beta t^2 w_t\mathbf{I}$$

3.3 Where is the Next Optimal Point?

Here we prove our main result of this section. We prove that over a long step the central path moves very little in directions orthogonal to the smallest eigenvector of the Hessian. We begin by noting the Hessian is approximately a scaled identity minus a rank 1 matrix.

LEMMA 3.4. *For all* t *we have*

$$\frac{1}{2}\left[t^2\cdot w_t\mathbf{I} - (t^2\cdot w_t - \mu_t)v_t v_t^\top\right] \preceq \nabla^2 f_t(x_t)$$

and

$$\nabla^2 f_t(x_t) \preceq t^2\cdot w_t\mathbf{I} - (t^2\cdot w_t - \mu_t)v_t v_t^\top$$

Using this and the lemmas of the previous section we bound the amount x_t can move in every direction far from v_t.

LEMMA 3.5 (ALMOST STRAIGHT PATH). *For all* $t \ge 0$, $\beta \in [0, \frac{1}{600}]$, *and any unit vector* y *with*

$$|\langle y, v_t\rangle| \le \frac{1}{t^2\cdot\kappa}$$

where

$$\kappa = \max_{\delta \in [t,(1+\beta)t]} \frac{w_\delta}{\mu_\delta}$$

we have

$$y^\top (x_{(1+\beta)t} - x_t) \leq \frac{6\beta}{t} \ .$$

3.4 Where is the End?

Here we bound the quality of the central path with respect to the geometric median objective. In particular, we show that if we can solve the problem for some $t = \frac{2n}{\epsilon f(x^*)}$ then we can obtain an $(1+\epsilon)$-approximate solution. As we will ultimately derive an algorithm that starting from initial $t = 1/O(f(x^*))$ and doubles t in every iteration, this will yield a $O(\log \frac{n}{\epsilon})$ iteration algorithm to obtain an $(1+\epsilon)$-approximate solution.

LEMMA 3.6. *For all $t > \frac{1}{400 f(x^*)}$, we have*

$$f(x_t) - f(x^*) \leq 100n\sqrt{\frac{f(x^*)}{t}} \ .$$

4. NEARLY LINEAR TIME GEOMETRIC MEDIAN

Here we show how to use the results from the previous section to obtain a nearly linear time algorithm for computing the geometric median. Our algorithm follows a simple structure (See Algorithm 1).

Algorithm 1: `AccurateMedian`(ϵ)

```
// Compute a 2-approximate geometric median and
   use it to center
```
$x^{(0)} :=$ `ApproximateMedian`(2)
Let $\widetilde{f}_* := f(x^{(0)})$, $t_i = \frac{1}{400\widetilde{f}_*}(1 + \frac{1}{600})^{i-1}$
$x^{(1)} =$ `LineSearch`$(x^{(0)}, t_1, t_1, 0, \epsilon_c)$ with $\epsilon_c = \frac{1}{10^{15} n^3 t_1^9 \cdot \widetilde{f}_*^3}$.

```
// Iteratively improve quality of approximation
for i ∈ [1, 1000 log(3000n/ε)] do
    // Compute ε_v-approximate minimum eigenvalue
       and eigenvector of ∇²f_{t_i}(x^(i))
```
$(\lambda^{(i)}, u^{(i)}) =$ `ApproxMinEig`$(x^{(i)}, t_i, \epsilon_v)$ with $\epsilon_v = \frac{1}{10^8 n^2 t_i^2 \cdot \widetilde{f}_*^2}$.

```
    // Line search to find x^(i+1) such that
       ‖x^(i+1) − x_{t_{i+1}}‖_2 ≤ ε_c / t_{i+1}
```
$x^{(i+1)} =$ `LineSearch`$(x^{(i)}, t_i, t_{i+1}, u^{(i)}, \epsilon_c)$ with $\epsilon_c = \frac{1}{10^{15} n^3 t_i^3 \cdot \widetilde{f}_*^3}$.

```
end
```
Output: ϵ-approximate geometric median $x^{(k)}$

First we use our result from Section A to compute an $(1+\epsilon_0)$-approximate median, denoted $x^{(1)}$. Then for a number of iterations we repeatedly move closer to x_t for some path parameter t, compute the minimum eigenvector of the Hessian, and line search in that direction to find an approximation to a point further along the central path. Ultimately, this yields a precise enough approximation to a point along the central path with large enough t that it is a high quality approximation to the geometric median.

We split the remainder of the algorithm specification and its analysis into several parts. First in Section 4.1 we show how to compute an approximate minimum eigenvector and eigenvalue of the Hessian of the penalized objective function. Then in Section 4.2 we show how to use this eigenvector to line search for the next central path point. Finally, in Section 4.3 we put these results together to obtain our nearly linear time algorithm. Throughout this section we will want an upper bound to $f(x_*)$ and will use $\widetilde{f}_*$ as this upper bound.

4.1 Eigenvector Computation and Hessian Approximation

Here we show how to compute the minimum eigenvector of $\nabla^2 f_t(x)$ and thereby obtain a concise approximation to $\nabla^2 f_t(x)$. Our main algorithmic tool is the well known power method and the fact that it converges quickly on a matrix with a large eigenvalue gap. We provide and analyze this method for completeness in Section C.3. Using this tool we estimate the top eigenvector as follows.

Algorithm 2: `ApproxMinEig`(x, t, ϵ)

Input: Point $x \in \mathbb{R}^d$, path parameter t, and target accuracy ϵ.
Let $\mathbf{A} = \sum_{i \in [n]} \frac{t^4 (x - a^{(i)})(x - a^{(i)})^\top}{(1 + g_t^{(i)}(y))^2 g_t^{(i)}(y)}$
Let $u :=$ `PowerMethod`$(\mathbf{A}, \Theta(\log(\frac{d}{\epsilon})))$
Let $\lambda = u^\top \nabla^2 f_t(x) u$
Output: (λ, u)

LEMMA 4.1 (HESSIAN EIGENVECTOR APPROXIMATION). *For any $x \in \mathbb{R}^d$, $t > 0$, and $\epsilon \in (0, \frac{1}{4})$, The algorithm* `ApproxMinEig`$(x, t, \epsilon)$ *outputs (λ, u) in $O(nd \log \frac{d}{\epsilon})$ time with high probability in d such that*

$$\langle v_t(x), u \rangle^2 \geq 1 - \epsilon$$

if $\mu_t(x) \leq \frac{1}{4} t^2 w_t(x)$. Furthermore, if

$$\epsilon \leq \frac{\mu_t(x)}{4t^2 \cdot w_t(x)} \leq \frac{1}{10^9 n^2 t^2 \cdot \widetilde{f}_*^2}$$

and

$$\mathbf{Q} \stackrel{\text{def}}{=} t^2 \cdot w_t(x) - \left(t^2 \cdot w_t(x) - \lambda\right) u u^\top$$

then $\frac{1}{4}\mathbf{Q} \preceq \nabla^2 f_t(x) \preceq 4\mathbf{Q}$.

Furthermore, we show that the $v^{(i)}$ computed by this algorithm is sufficiently close to the bad direction.Using this lemma we obtain the following. Using Lemma C.4, a minor technical lemma regarding the transitivity of large inner products, yields:

LEMMA 4.2. *Let $u =$ `ApproxMinEig`(x, t, ϵ_v) for some x such that $\|x - x_t\|_2 \leq \frac{\epsilon_c}{t}$ for $\epsilon_c \leq \frac{1}{10^6}$. Suppose that $\mu_t \leq \frac{1}{4} t^2 \cdot w_t$. Then, for all unit vectors $y \perp u$, we have*

$$\langle y, v_t \rangle^2 \leq \max\{500\epsilon_c^{2/3}, 10\epsilon_v\} \ .$$

Note that the lemma above assumed μ_t is small. For the case μ_t is large, we show that the next central path point is close to the current point and hence we do not need to know the bad direction.

LEMMA 4.3. *Suppose that* $\mu_t \geq \frac{1}{4}t^2 \cdot w_t$. *Then, we have* $\|x_s - x_t\|_2 \leq \frac{1}{100t}$ *for all* $s \in [1, 1.001t]$.

4.2 Line Searching

Here we show how to line search along the bad direction to find the next point on the central path. Unfortunately, it is not clear if you can binary search on the objective function directly. It might be the case that if we searched over α to minimize $f_{t_{i+1}}(y^{(i)} + \alpha v^{(i)})$ we might obtain a point far away from x_{t+1} and therefore be unable to move towards x_{t+1} efficiently.

To overcome this difficulty, we use the fact that over the region $\|x - y\|_2 = O(\frac{1}{t})$ the Hessian changes by at most a constant and therefore we can minimize $f_t(x)$ over this region extremely quickly. Therefore, we instead line search on the following function

$$g_{t,y,v}(\alpha) \stackrel{\text{def}}{=} \min_{\|x-(y+\alpha v)\|_2 \leq \frac{1}{100t}} f_t(x) \tag{3}$$

and use that we can evaluate $g_{t,y,v}(\alpha)$ approximately by using an appropriate centering procedure. We can show (See Lemma C.5) that $g_{t,y,v}(\alpha)$ is convex and therefore we can minimize it efficiently just by doing an appropriate binary search. By finding the approximately minimizing α and outputting the corresponding approximately minimizing x, we can obtain $x^{(i+1)}$ that is close enough to $x_{t_{i+1}}$. For notational convenience, we simply write $g(\alpha)$ if t, y, v is clear from the context.

First, we show how we can locally center and provide error analysis for that algorithm.

Algorithm 3: `LocalCenter`(y, t, ϵ)

Input: Point $y \in \mathbb{R}^d$, path parameter t, target accuracy ϵ.
Let $(\lambda, v) :=$ `ApproxMinEig`$(x, t, 10^{-9}n^{-2}t^{-2} \cdot \widetilde{f}_*^{-2})$.
Let $\mathbf{Q} = t^2 \cdot w_t(y)\mathbf{I} - \left(t^2 \cdot w_t(y) - \lambda\right) vv^\top$
Let $x^{(0)} = y$
for $i = 1, ..., k = 64 \log \frac{1}{\epsilon}$ **do**
  Let $x^{(i)} = \min_{\|x-y\|_2 \leq \frac{1}{100t}} f(x^{(i-1)}) + \langle \nabla f_t(x^{(i-1)}), x - x^{(i-1)} \rangle + 4\|x - x^{(i-1)}\|_{\mathbf{Q}}^2$.
end
Output: $x^{(k)}$

LEMMA 4.4. *Given some* $y \in \mathbb{R}^d$, $t > 0$ *and* $\epsilon > 0$. *In* $O(nd \log(\frac{nt \cdot \widetilde{f}_*}{\epsilon}))$ *time with high probability,* `LocalCenter`(y, t, ϵ) *computes* $x^{(k)}$ *such that*

$$f_t(x^{(k)}) - \min_{\|x-y\|_2 \leq \frac{1}{100t}} f_t(x) \leq \epsilon \left(f_t(y) - \min_{\|x-y\|_2 \leq \frac{1}{100t}} f_t(x) \right).$$

Using this local centering algorithm as well as a general result for minimizing one dimensional convex functions using a noisy oracle (See Section C.4) we obtain our line search algorithm.

LEMMA 4.5. *Given* x *such that* $\|x - x_t\|_2 \leq \frac{\epsilon_c}{t}$ *with* $t \geq \frac{1}{400 f(x^*)}$ *and* $\epsilon_c \leq \frac{1}{10^{15} n^3 t^3 \cdot \widetilde{f}_*^3}$. *Let* $u =$ `ApproxMinEig`(x, t, ϵ_v) *with* $\epsilon_v \leq \frac{1}{10^8 n^2 t^2 \cdot \widetilde{f}_*^2}$. *Then, in* $O(nd \log^2(\frac{nt \cdot \widetilde{f}_*}{\epsilon}))$ *time, the algorithm* `LineSearch`(x, t, t', u, ϵ) *output* y *such that* $\|y - x_{t'}\|_2 \leq \frac{\epsilon}{t'}$.

Algorithm 4: `LineSearch`(x, t, t', u, ϵ)

Input: Point $x \in \mathbb{R}^d$, current path parameter t, next path parameter t', bad direction u, target accuracy ϵ
Let $\epsilon_O = \frac{\epsilon^2}{10^{10} t^3 n^3 \cdot \widetilde{f}_*^3}$, $\ell = -12\widetilde{f}_*$, $u = 12\widetilde{f}_*$.
Define the oracle $q : \mathbb{R} \to \mathbb{R}$ by
$q(\alpha) = f_{t'}(\texttt{LocalCenter}(x + \alpha u, t', \epsilon_O))$
Let $\alpha' =$ `OneDimMinimizer`$(\ell, u, \epsilon_O, q, tn)$
Output: $x' =$ `LocalCenter`$(x + \alpha u, t', \epsilon_O)$

We also provide the following lemma useful for finding the first center.

LEMMA 4.6. *Given* x *such that* $\|x - x_t\|_2 \leq \frac{1}{100t}$ *with* $t \geq \frac{1}{400 f(x^*)}$. *Then, in* $O(nd \log^2(\frac{nt \cdot \widetilde{f}_*}{\epsilon}))$ *time, the algorithm* `LineSearch`(x, t, t, u, ϵ) *outputs* y *such that* $\|y - x_t\|_2 \leq \frac{\epsilon}{t}$ *for any vector* u.

4.3 Putting It All Together

Here we show how to put together the results of the previous sections to prove our main theorem.

THEOREM 4.7. *In* $O(nd \log^3(\frac{n}{\epsilon}))$ *time, Algorithm 1 outputs an* $(1+\epsilon)$*-approximate geometric median with constant probability.*

5. ACKNOWLEDGMENTS

We thank Yan Kit Chim, Ravi Kannan, and Jonathan A. Kelner for many helpful conversations. We thank the reviewers for their help in completing the previous work table. This work was partially supported by NSF awards 0843915, 1065106 and 1111109, NSF Graduate Research Fellowship (grant no. 1122374) and Sansom Graduate Fellowship in Computer Science. Part of this work was done while authors were visiting the Simons Institute for the Theory of Computing, UC Berkeley.

6. REFERENCES

[1] M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 250–257, 2002.

[2] C. Bajaj. The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3(2):177–191, 1988.

[3] E. Balas and C.-S. Yu. A note on the weiszfeld-kuhn algorithm for the general fermat problem. *Managme Sci Res Report*, (484):1–6, 1982.

[4] P. Bose, A. Maheshwari, and P. Morin. Fast approximations for sums of distances, clustering and the Fermat-Weber problem. *Computational Geometry*, 24(3):135 – 146, 2003.

[5] S. Bubeck. Theory of convex optimization for machine learning. *arXiv preprint arXiv:1405.4980*, 2014.

[6] R. Chandrasekaran and A. Tamir. Open questions concerning weiszfeld's algorithm for the fermat-weber location problem. *Mathematical Programming*, 44(1-3):293–295, 1989.

[7] H. H. Chin, A. Madry, G. L. Miller, and R. Peng. Runtime guarantees for regression problems. In *ITCS*, pages 269–282, 2013.

[8] L. Cooper and I. Katz. The weber problem revisited. *Computers and Mathematics with Applications*, 7(3):225 – 234, 1981.

[9] Z. Drezner, K. Klamroth, A. SchÃűbel, and G. Wesolowsky. *Facility location*, chapter The Weber problem, pages 1–36. Springer, 2002.

[10] D. Feldman and M. Langberg. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578. ACM, 2011.

[11] C. C. Gonzaga. Path-following methods for linear programming. *SIAM review*, 34(2):167–224, 1992.

[12] S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 126–134. ACM, 2005.

[13] P. Indyk and S. U. C. S. Dept. *High-dimensional computational geometry*. Stanford University, 2000.

[14] J. Krarup and S. Vajda. On torricelli's geometrical solution to a problem of fermat. *IMA Journal of Management Mathematics*, 8(3):215–224, 1997.

[15] R. A. Kronmal and A. V. Peterson. The alias and alias-rejection-mixture methods for generating random variables from probability distributions. In *Proceedings of the 11th Conference on Winter Simulation - Volume 1*, WSC '79, pages 269–280, Piscataway, NJ, USA, 1979. IEEE Press.

[16] H. Kuhn. A note on fermat's problem. *Mathematical Programming*, 4(1):98–107, 1973.

[17] Y. T. Lee and A. Sidford. Path-finding methods for linear programming : Solving linear programs in õ(sqrt(rank)) iterations and faster algorithms for maximum flow. In *55th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2014, 18-21 October, 2014, Philadelphia, PA, USA*, pages 424–433, 2014.

[18] H. P. Lopuhaa and P. J. Rousseeuw. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *Ann. Statist.*, 19(1):229–248, 03 1991.

[19] H. P. Lopuhaa and P. J. Rousseeuw. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, pages 229–248, 1991.

[20] A. Madry. Navigating central path with electrical flows: from flows to matchings, and back. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science*, 2013.

[21] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume I. 2003.

[22] Y. Nesterov and A. S. Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Society for Industrial and Applied Mathematics, 1994.

[23] L. M. Ostresh. On the convergence of a class of iterative methods for solving the weber location problem. *Operations Research*, 26(4):597–609, 1978.

[24] P. A. Parrilo and B. Sturmfels. Minimizing polynomial functions. In *DIMACS Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science, March 12-16, 2001, DIMACS Center, Rutgers University, Piscataway, NJ, USA*, pages 83–100, 2001.

[25] F. Plastria and M. Elosmani. On the convergence of the weiszfeld algorithm forăcontinuous single facility location allocation problems. *TOP*, 16(2):388–406, 2008.

[26] J. Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Mathematical Programming*, 40(1-3):59–93, 1988.

[27] Y. Vardi and C.-H. Zhang. The multivariate l1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4):1423–1426, 2000.

[28] V. Viviani. De maximis et minimis geometrica divinatio liber 2. *De Maximis et Minimis Geometrica Divinatio*, 1659.

[29] A. Weber. *The Theory of the Location of Industries*. Chicago University Press, 1909. Aber den I der Industrien.

[30] E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnes est minimum. *Tohoku Mathematical Journal*, pages 355–386, 1937.

[31] G. Xue and Y. Ye. An efficient algorithm for minimizing a sum of euclidean norms with applications. *SIAM Journal on Optimization*, 7:1017–1036, 1997.

[32] Y. Ye. *Interior point algorithms: theory and analysis*, volume 44. John Wiley & Sons, 2011.

APPENDIX

A. PSEUDOPOLYNOMIAL ALGORITHM

Here we provide a self-contained result on computing a $1+\epsilon$ approximate geometric median in $O(d\epsilon^{-2})$ time. Note that it is impossible to achieve such approximation for the mean, $\min_{x\in\mathbb{R}^d}\sum_{i\in[n]}\|x-a^{(i)}\|_2^2$, because the mean can be changed arbitrarily by changing only 1 point. However, [19] showed that the geometric median is far more stable. In Section A.1, we show how this stability property allows us to get an constant approximate in $O(d)$ time. In Section A.2, we show how to use stochastic subgradient descent to then improve the accuracy.

A.1 A Constant Approximation of Geometric Median

We first prove that the geometric median is stable even if we are allowed to modify up to half of the points. The following lemma is a strengthening of the robustness result in [19].

LEMMA A.1. *Let x^* be a geometric median of $\{a^{(i)}\}_{i\in[n]}$ and let $S\subseteq[n]$ with $\|S\|<\frac{n}{2}$. For all x*

$$\|x^*-x\|_2\le\left(\frac{2n-2|S|}{n-2|S|}\right)\max_{i\notin S}\|a^{(i)}-x\|_2\,.$$

PROOF. For notational convenience let $r=\|x^*-x\|_2$ and let $M=\max_{i\notin S}\|a^{(i)}-x\|_2$.

For all $i\notin S$, we have that $\|x-a^{(i)}\|_2\le M$, hence, we

have

$$\begin{aligned}\|x^* - a^{(i)}\|_2 &\geq r - \|x - a^{(i)}\|_2 \\ &\geq r - 2M + \|x - a^{(i)}\|_2 .\end{aligned}$$

Furthermore, by triangle inequality for all $i \in S$, we have

$$\|x^* - a^{(i)}\|_2 \geq \|x - a^{(i)}\|_2 - r .$$

Hence, we have that

$$\sum_{i\in[n]} \|x^* - a^{(i)}\|_2 \geq \sum_{i\in[n]} \|x - a^{(i)}\|_2 + (n - |S|)(r - 2M) - |S|r .$$

Since x^* is a minimizer of $\sum_i \|x^* - a^{(i)}\|_2$, we have that

$$(n - |S|)(r - 2M) - |S|r \leq 0.$$

Hence, we have

$$\|x^* - x\|_2 = r \leq \frac{2n - 2|S|}{n - 2|S|} M.$$

□

Now, we use Lemma A.1 to show that `CrudeApproximate` outputs a constant approximation of the geometric median with high probability.

Algorithm 5: $\texttt{CrudeApproximate}_K$

Input: $a^{(1)}, a^{(2)}, \cdots, a^{(n)} \in \mathbb{R}^d$.
Sample two independent random subset of $[n]$ of size K. Call them S_1 and S_2.
Let $i^* \in \arg\min_{i\in S_2} \alpha_i$ where α_i is the 65 percentile of the numbers $\{\|a^{(i)} - a^{(j)}\|_2\}_{j\in S_1}$.
Output: Output $a^{(i^*)}$ and α_{i^*}.

LEMMA A.2. *Let x^* be a geometric median of $\{a^{(i)}\}_{i\in[n]}$ and $(\widetilde{x}, \lambda)$ be the output of $\texttt{CrudeApproximate}_K$. We define $d_T^k(x)$ be the k-percentile of $\left\{\|x - a^{(i)}\|\right\}_{i\in T}$. Then, we have that $\|x^* - \widetilde{x}\|_2 \leq 6d_{[n]}^{60}(\widetilde{x})$. Furthermore, with probability $1 - e^{-\Theta(K)}$, we have*

$$d_{[n]}^{60}(\widetilde{x}) \leq \lambda = d_{S_1}^{60}(\widetilde{x}) \leq 2d_{[n]}^{70}(x^*).$$

PROOF. Lemma A.1 shows that for all x and $T \subseteq [n]$ with $|T| \leq \frac{n}{2}$

$$\|x^* - x\|_2 \leq \left(\frac{2n - 2|T|}{n - 2|T|}\right) \max_{i\notin T} \|a^{(i)} - x\|_2 .$$

Picking T to be the indices of largest 40% of $\|a^{(i)} - \widetilde{x}\|_2$, we have

$$\|x^* - \widetilde{x}\|_2 \leq \left(\frac{2n - 0.8n}{n - 0.8n}\right) d_{[n]}^{60}(\widetilde{x}) = 6d_{[n]}^{60}(\widetilde{x}). \tag{4}$$

For any point x, we have that $d_{[n]}^{60}(x) \leq d_{S_1}^{65}(x)$ with probability $1 - e^{-\Theta(K)}$ because S_1 is a random subset of $[n]$ with size K. Taking union bound over elements on S_2, with probability $1 - Ke^{-\Theta(K)} = 1 - e^{-\Theta(K)}$, for all points $x \in S_2$

$$d_{[n]}^{60}(x) \leq d_{S_1}^{65}(x). \tag{5}$$

yielding that $d_{[n]}^{60}(\widetilde{x}) \leq \lambda$.

Next, for any $i \in S_2$, we have

$$\|a^{(i)} - a^{(j)}\|_2 \leq \|a^{(i)} - x^*\|_2 + \|x^* - a^{(j)}\|_2.$$

and hence

$$d_{[n]}^{70}(a^{(i)}) \leq \|a^{(i)} - x^*\|_2 + d_{[n]}^{70}(x^*).$$

Again, since S_1 is a random subset of $[n]$ with size K, we have that $d_{S_1}^{65}(a^{(i)}) \leq d_{[n]}^{70}(a^{(i)})$ with probability $1 - Ke^{-\Theta(K)} = 1 - e^{-\Theta(K)}$. Therefore,

$$d_{S_1}^{65}(a^{(i)}) \leq \|a^{(i)} - x^*\|_2 + d_{[n]}^{70}(x^*).$$

Since S_2 is an independent random subset, with probability $1 - e^{-\Theta(K)}$, there is $i \in S_2$ such that $\|a^{(i)} - x^*\|_2 \leq d_{[n]}^{70}(x^*)$. In this case, we have

$$d_{S_1}^{65}(a^{(i)}) \leq 2d_{[n]}^{70}(x^*).$$

Since i^* minimize $d_{S_1}^{65}(a^{(i)})$ over all $i \in S_2$, we have that

$$\lambda \stackrel{\text{def}}{=} d_{S_1}^{65}(\widetilde{x}) \stackrel{\text{def}}{=} d_{S_1}^{65}(a^{(i^*)}) \leq d_{S_1}^{65}(a^{(i)}) \leq 2d_{[n]}^{70}(x^*) .$$

□

A.2 A $(1+\epsilon)$-Approximate Geometric Median

Here we show how to improve the constant approximation in the previous section to a $1 + \epsilon$ approximation. Our algorithm is essentially stochastic subgradient where we use the information from the previous section to bound the domain in which we need to search for a geometric median.

Algorithm 6: $\texttt{ApproximateMedian}(\epsilon)$

Input: $a^{(1)}, a^{(2)}, \cdots, a^{(n)} \in \mathbb{R}^d$.
Let $T = (120/\epsilon)^2$ and let $\eta = \frac{6\lambda}{n}\sqrt{\frac{2}{T}}$.
Let
$(x^{(1)}, \lambda) = \texttt{CrudeApproximate}_{\sqrt{T}}(a^{(1)}, a^{(2)}, \cdots, a^{(n)})$.
for $k \leftarrow 1, 2, \cdots, T$ **do**
  Sample i_k from $[n]$ and let
  $g^{(k)} =$
  $\begin{cases} n(x^{(k)} - a^{(i_k)})/\|x^{(k)} - a^{(i_k)}\|_2 & \text{if } x^{(i)} \neq a^{(i_k)} \\ 0 & \text{otherwise} \end{cases}$
  Let $x^{(k+1)} =$
  $\arg\min_{\|x - x^{(1)}\|_2 \leq 6\lambda} \eta \left\langle g^{(k)}, x - x^{(k)} \right\rangle + \frac{1}{2}\|x - x^{(k)}\|_2^2$.
end
Output: Output $\frac{1}{T}\sum_{i=1}^T x^{(k)}$.

THEOREM A.3. *With probability $1 - e^{-\Theta(1/\epsilon)}$ the output, x, of $\texttt{ApproximateMedian}(\epsilon)$ satisfies*

$$\mathbb{E} f(x) \leq (1 + \epsilon) \min_{x\in\mathbb{R}^d} f(x).$$

Furthermore, the algorithm takes $O(d/\epsilon^2)$ time.

PROOF. After computing $x^{(1)}$ and λ the remainder of our algorithm is the stocastic subgradient descent method applied to $f(x)$. It is routine to check that $\mathbb{E}_{i^{(k)}} g^{(k)}$ is a subgradient of f at $x^{(k)}$. Furthermore, since the diameter of the domain, $\left\{x : \|x - x^{(1)}\|_2 \leq 6\lambda\right\}$, is clearly λ and the norm of sampled gradient, $g^{(k)}$, is at most n, we have that

$$\mathbb{E} f\left(\frac{1}{T}\sum_{i=1}^T x^{(k)}\right) - \min_{\|x - x^{(1)}\|_2 \leq 6\lambda} f(x) \leq 6n\lambda\sqrt{\frac{2}{T}}$$

(see [5, Thm 6.1]). Lemma A.2 shows that $\|x^* - x^{(1)}\|_2 \leq 6\lambda$ and $\lambda \leq 2d^{70}_{[n]}(x^*)$ with probability $1 - \sqrt{T}e^{-\Theta(\sqrt{T})}$. In this case, we have

$$\mathbb{E}f\left(\frac{1}{T}\sum_{i=1}^{T} x^{(k)}\right) - f(x^*) \quad \leq \quad \frac{12\sqrt{2}nd^{70}_{[n]}(x^*)}{\sqrt{T}}.$$

Since $d^{70}_{[n]}(x^*) \leq \frac{1}{0.3n}f(x^*)$, we have

$$\mathbb{E}f\left(\frac{1}{T}\sum_{i=1}^{T} x^{(k)}\right) \quad \leq \quad \left(1 + \frac{60}{\sqrt{T}}\right)f(x^*) \leq \left(1 + \frac{\epsilon}{2}\right)f(x^*)\,.$$

□

B. DERIVATION OF PENALTY FUNCTION

Here in the full version we derive our penalized objective function. Consider the following optimization problem.

$$\min_{x\in\mathbb{R}^d, \alpha \geq 0 \in \mathbb{R}^n} f_t(x, \alpha)$$

where

$$f_t(x,\alpha) = t \cdot 1^T\alpha + \sum_{i\in[n]} -\ln\left(\alpha_i^2 - \|x - a^{(i)}\|_2^2\right) \ .$$

As, $-\ln(\alpha_i^2 - \|x - a^{(i)}\|_2^2)$ is a natural barrier function for the set $\alpha_i^2 \geq \|x - a^{(i)}\|_2^2$ we see that as we minimize this function for increasing values of t the x values converge to a solution to the geometric median problem.

The penalty function we use $f_t(x)$ is simply this function after we solve for the optimal α_i explicitly. To see this fix and x and t. Note that for all $j \in [n]$ we have

$$\frac{\partial}{\partial\alpha_j}f(x,\alpha) = t - \left(\frac{1}{\alpha_j^2 - \|x - a^{(i)}\|_2^2}\right)2\alpha_j$$

and therefore, as $f(x,\alpha)$ is convex in α, the minimum α_j^* must satisfy

$$t\left(\left(\alpha_j^*\right)^2 - \|x - a^{(i)}\|_2^2\right) - 2\alpha_j^* = 0 \ .$$

Solving for such α_j^* we get

$$\alpha_j^* = \frac{2 + \sqrt{4 + 4t^2\|x - a^{(i)}\|_2^2}}{2t} = \frac{1}{t}\left[1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}\right] \ .$$

and consequently

$$\begin{aligned}\left(\alpha_j^*\right)^2 &= \frac{1}{t^2}\left[1 + 2\sqrt{1 + t^2\|x - a^{(i)}\|_2^2} + 1 + t^2\|x - a^{(i)}\|_2^2\right] \\ &= \frac{2}{t^2}\left[1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}\right] + \|x - a^{(i)}\|_2^2 \ .\end{aligned}$$

Thus we can write the problem as

$$\min_{x\in\mathbb{R}^d} \sum_{i\in[n]} \left[1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2} - \ln\left[\frac{2}{t^2}\left(1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}\right)\right]\right] \ .$$

If we drop the constants, we get our penalty function f_t :

$$\min_{x\in\mathbb{R}^d} f_t(x)$$

where

$$f_t(x) = \sum_{i\in[n]} \sqrt{1 + t^2\|x - a^{(i)}\|_2^2} - \ln\left[1 + \sqrt{1 + t^2\|x - a^{(i)}\|_2^2}\right].$$

C. TECHNICAL FACTS

Here in we provide various technical lemmas we use through the paper.

C.1 General Math

The following lemma shows that given any matrix obtained as a convex combination of the identity minus a rank 1 matrix less than the identity results in a matrix that is well approximation spectrally by the identity minus a rank 1 matrix. We use this Lemma to characterize the Hessian of our penalized objective function and thereby imply that it is possible to apply the inverse of the Hessian to a vector with high precision.

LEMMA C.1. *Given any matrix*

$$\mathbf{A} = \sum_i \left(\alpha_i\mathbf{I} - \beta_i a_i a_i^\top\right) \in \mathbb{R}^{d\times d}$$

where a_i are unit vectors and $0 \leq \beta_i \leq \alpha_i$ for all i. We have that

$$\frac{1}{2}\left(\sum_i \alpha_i\mathbf{I} - \lambda vv^\top\right) \preceq \mathbf{A} \preceq \sum_i \alpha_i\mathbf{I} - \lambda vv^\top$$

where v is a unit vector that is the maximum eigenvector of $\sum_i \beta_i a_i a_i^\top$ and λ is the corresponding eigenvalue.

PROOF. Let $\lambda_1 \geq ... \geq \lambda_d$ denote the eigenvalues of $\sum_i \beta_i a_i a_i^\top$. Clearly

$$\begin{aligned}\mathrm{tr}(\mathbf{A}) &= \sum_i \lambda_i = d\sum_i \alpha_i - \sum_i \beta_i \\ &\geq (d-1)\sum_i \alpha_i\end{aligned}$$

Also clearly, $v^\top\mathbf{A}v = v^\top\left(\sum_i \alpha_i\mathbf{I} - \lambda vv^\top\right)v$, and therefore it simply remains to show that $\lambda_j \in [\frac{1}{2}\sum_i \alpha_i, \sum_i \alpha_i]$ for all $j > 1$. Since, $\lambda_j \leq \sum_i \alpha_i$ for all j, we have that

$$\begin{aligned}(d-1)\sum_i \alpha_i \leq \mathrm{tr}(\mathbf{A}) = \sum_i \lambda_i &< 2\lambda_2 + \sum_{j\geq 3}\lambda_i \\ &\leq 2\lambda_2 + (d-1)\sum_i \alpha_i.\end{aligned}$$

Therefore, $\lambda_j \geq \lambda_2 \geq \frac{1}{2}\sum_i \alpha_i$ for all $j > 1$. □

Next we show how to bound the difference between the outer product of two unit vectors by their inner product. We use this lemma to bound the amount of precision required in our eigenvector computations.

LEMMA C.2. *For unit vectors u_1 and u_2 we have*

$$\|u_1u_1^\top - u_2u_2^\top\|_2^2 = 1 - (u_1^\top u_2)^2 \tag{6}$$

Consequently if $\left(u_1^\top u_2\right)^2 \geq 1 - \epsilon$ for $\epsilon \leq 1$ we have that

$$-\sqrt{\epsilon}\mathbf{I} \preceq u_1u_1^\top - u_2u_2^\top \preceq \sqrt{\epsilon}\mathbf{I}$$

PROOF. Note that $u_1u_1^\top - u_2u_2^\top$ is a symmetric matrix and all eigenvectors are either orthogonal to both u_1 and u_2 (with eigenvalue 0) or are of the form $v = \alpha u_1 + \beta u_2$ where α and β are real numbers that are not both 0. Thus, if v is an eigenvector of eigenvalue λ it must be that

$$\begin{aligned}\gamma(\alpha u_1 + \beta u_2) &= \left(u_1u_1^\top - u_2u_2^\top\right)(\alpha u_1 + \beta u_2) \\ &= (\alpha + \beta(u_1^\top u_2))u_1 - (\alpha(u_1^\top u_2) + \beta)u_2\end{aligned}$$

Consequently

$$\begin{pmatrix} (1-\gamma) & u_1^\top u_2 \\ -(u_1^\top u_2) & -(1+\gamma) \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

By computing the determinant we see this has a solution only when

$$-(1-\gamma^2) + (u_1^\top u_2)^2 = 0$$

Solving for γ then yields (6) and completes the proof. □

Next we show how the top eigenvectors of two spectrally similar matrices are related. We use this to bound the amount of spectral approximation we need to obtain accuracate eigenvector approximations.

LEMMA C.3. *Let $\mathbf{A}$ be a PSD matrix such that $(1-\epsilon)\mathbf{A} \preceq \mathbf{B} \preceq (1+\epsilon)\mathbf{A}$. Let $g \stackrel{\text{def}}{=} \frac{\lambda_1(\mathbf{A})-\lambda_2(\mathbf{A})}{\lambda_1(\mathbf{A})}$ where $\lambda_1(\mathbf{A})$ and $\lambda_2(\mathbf{A})$ are the largest and second largest eigenvalue of $\mathbf{A}$. Then, we have*

$$\left(v_1(\mathbf{A})^\top v_1(\mathbf{B})\right)^2 \geq 1 - 3\frac{\epsilon}{g}.$$

PROOF. Without loss of generality $v_1(\mathbf{B}) = \alpha v_1(\mathbf{A}) + \beta v$ for some unit vector $v \perp v_1(\mathbf{A})$ and $\alpha, \beta \in \mathbb{R}$ such that $\alpha^2 + \beta^2 = 1$. Now we know that

$$\begin{aligned}v_1(\mathbf{B})^\top \mathbf{B} v_1(\mathbf{B}) &\leq (1+\epsilon)v_1(\mathbf{B})^\top \mathbf{A} v_1(\mathbf{B}) \\ &\leq (1+\epsilon)\left[\alpha^2\lambda_1(\mathbf{A}) + \beta^2\lambda_2(\mathbf{A})\right] .\end{aligned}$$

Furthermore, by the optimality of $v_1(\mathbf{B})$ we have that

$$v_1(\mathbf{B})^\top \mathbf{B} v_1(\mathbf{B}) \geq (1-\epsilon)v_1(\mathbf{A})^\top \mathbf{A} v_1(\mathbf{A}) \geq (1-\epsilon)\lambda_1(\mathbf{A}).$$

Now since $\beta^2 = 1 - \alpha^2$ we have that

$$(1-\epsilon)\lambda_1(\mathbf{A}) \leq (1+\epsilon)\alpha^2\left(\lambda_1(\mathbf{A}) - \lambda_2(\mathbf{A}) + (1+\epsilon)\lambda_2(\mathbf{A})\right).$$

Thus we have that

$$\alpha^2 \geq \frac{\lambda_1(\mathbf{A}) - \lambda_2(\mathbf{A}) - \epsilon(\lambda_1(\mathbf{A}) + \lambda_2(\mathbf{A}))}{(1+\epsilon)(\lambda_1(\mathbf{A}) - \lambda_2(\mathbf{A}))}$$

Since $\frac{1}{1+\epsilon} \geq 1-\epsilon$, $\lambda_2(\mathbf{A}) \leq \lambda_1(\mathbf{A})$, and $g \leq 1$ we have

$$\begin{aligned}\alpha^2 &\geq \left(1 - 2\epsilon\left(\frac{\lambda_1(\mathbf{A})}{\lambda_1(\mathbf{A}) - \lambda_2(\mathbf{A})}\right)\right)(1-\epsilon) \\ &= 1 - 2\frac{\epsilon}{g} - \epsilon + 2\frac{\epsilon^2}{g} \geq 1 - 3\frac{\epsilon}{g}.\end{aligned}$$

□

Here we prove a an approximate transitivity lemma for inner products of vectors. We use this to bound the accuracy need for certain eigenvector computations.

LEMMA C.4. *Suppose that we have vectors $v_1, v_2, v_3 \in \mathbb{R}^n$ such that $\langle v_1, v_2\rangle^2 \geq 1-\epsilon$ and $\langle v_2, v_3\rangle^2 \geq 1-\epsilon$ for $0 < \epsilon \leq \frac{1}{4}$ then $\langle v_1, v_3\rangle^2 \geq 1 - 4\epsilon$.*

PROOF. Without loss of generality, we can write $v_1 = \alpha_1 v_2 + \beta_1 w_1$ for $\alpha_1^2 + \beta_1^2 = 1$ and unit vector $w_1 \perp v_2$. Similarly we can write $v_3 = \alpha_3 v_2 + \beta_3 w_3$ for $\alpha_3^2 + \beta_3^2 = 1$ and unit vector $w_3 \perp v_2$. Now, by the inner products we know that $\alpha_1^2 \geq 1-\epsilon$ and $\alpha_3^2 \geq 1-\epsilon$ and therefore $|\beta_1| \leq \sqrt{\epsilon}$ and $|\beta_3| \leq \sqrt{\epsilon}$. Consequently, since $\epsilon \leq \frac{1}{4}$, we have that

$$\begin{aligned}\langle v_1, v_3\rangle^2 &\geq \langle \alpha_1 v_2 + \beta_1 w_1, \alpha_3 v_2 + \beta_3 w_3\rangle^2 \geq (\alpha_1\alpha_3 - |\beta_1\beta_3|)^2 \\ &\geq (1 - \epsilon - \epsilon)^2 \geq (1-2\epsilon)^2 \geq 1 - 4\epsilon.\end{aligned}$$

□

C.2 Convex Optimization

Here we provide a short technical lemma about the convexity of functions that arises naturally in our line searching procedure.

LEMMA C.5. *Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ be a convex function and and let $g(\alpha) \stackrel{\text{def}}{=} \min_{x\in S} f(x + \alpha d)$ for any convex set S and $d \in \mathbb{R}^n$. Then g is convex.*

PROOF. Let $\alpha, \beta \in \mathbb{R}$ and define $x_\alpha = \arg\min_{x\in S} f(x + \alpha d)$ and $x_\beta = \arg\min_{x\in S} f(x + \beta d)$. For any $t \in [0,1]$ we have

$$\begin{aligned}g(t\alpha + (1-t)\beta) &= \min_{x\in S} f(x + (t\alpha + (1-t)\beta) \\ &\leq f(tx_\alpha + (1-t)x_\beta + (t\alpha + (1-t)\beta)d) \\ &\qquad \text{(Convexity of } S\text{)} \\ &\leq t\cdot f(x_\alpha + \alpha d) + (1-t)\cdot f(x_\beta + \beta\cdot d) \\ &\qquad \text{(Convexity of } f\text{)} \\ &= t\cdot g(\alpha) + (1-t)\cdot g(\beta)\end{aligned}$$

□

LEMMA C.6. *For any vectors $y, z, v \in \mathbb{R}^d$ and scalar α, we can minimize $\min_{\|x-y\|_2^2 \leq \alpha} \|x - z\|^2_{I - vv^\top}$ exactly in time $O(d)$.*

PROOF. Let x^* be the solution of this problem. If $\|x^* - y\|_2 < \alpha$, then $x^* = z$. Otherwise, there is $\lambda > 0$ such that x^* is the minimizer of

$$\min_x \|x - z\|^2_{I-vv^\top} + \lambda\|x - y\|^2.$$

Let $Q = I - vv^\top$. Then, the optimality condition shows that

$$Q(x - z) + \lambda(x - y) = 0.$$

Therefore, we have

$$x = (Q + \lambda I)^{-1}(Qz + \lambda y). \tag{7}$$

Hence, we have

$$\alpha = \|x - y\|_2^2 = (z-y)^\top Q(Q + \lambda I)^{-2}Q(z - y).$$

Let $\eta = 1 + \lambda$, then we have $(Q + \lambda I) = \eta I - vv^\top$ and hence Sherman–Morrison formula shows that

$$\begin{aligned}(Q + \lambda I)^{-1} &= \eta^{-1}I + \frac{\eta^{-2}vv^\top}{1 - \|v\|^2\eta^{-1}} \\ &= \eta^{-1}\left(I + \frac{vv^\top}{\eta - \|v\|^2}\right).\end{aligned}$$

Hence, we have

$$\begin{aligned}(Q + \lambda I)^{-2} &= \eta^{-2}\left(I + \frac{2vv^\top}{\eta - \|v\|^2} + \frac{vv^\top\|v\|^2}{(\eta - \|v\|^2)^2}\right) \\ &= \eta^{-2}\left(I + \frac{2\eta - \|v\|^2}{(\eta - \|v\|^2)^2}vv^\top\right)\end{aligned}$$

Let $c_1 = \|Q(z-y)\|_2^2$ and $c_2 = \left(v^\top Q(z-y)\right)^2$, then we have

$$\alpha \;=\; \eta^{-2}\left(c_1 + \frac{2\eta - \|v\|^2}{\left(\eta - \|v\|^2\right)^2} c_2\right).$$

Hence, we have

$$\alpha\eta^2\left(\eta - \|v\|^2\right)^2 = c_1\left(\eta - \|v\|^2\right)^2 + c_2\left(2\eta - \|v\|^2\right).$$

Note that this is a polynomial of degree 4 in η and all coefficients can be computed in $O(d)$ time. Solving this by explicit formula, one can test all 4 possible η's into the formula (7) of x. Together with trivial case $x^* = z$, we simply need to check among 5 cases to check which is the solution. □

C.3 Power Method

Here for completeness we prove the correctness of the well-known power method which we use to approximate the Hessian of our penalized objective function.

Algorithm 7: `PowerMethod`$(\mathbf{A}, k)$

Input: a positive definite matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$, and the number of iterations k.
Let $x \sim \mathcal{N}(0,1)$ be drawn from a d dimensional normal distribution.
Let $y = \mathbf{A}^k z$
Output: $u = y/\|y\|_2$

Lemma C.7 (Power Method). *Let $\mathbf{A} \in \mathbb{R}^{d\times d}$ is a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d \geq 0$, corresponding eigenvectors $v_1, ..., v_d \in \mathbb{R}^d$, and $g \stackrel{\text{def}}{=} \frac{\lambda_1 - \lambda_2}{\lambda_1}$. For any $\epsilon > 0$ and $k \geq \frac{\alpha}{g}\log(\frac{d}{\epsilon})$ for large enough constant α, in time $O(\text{nnz}(\mathbf{A})\cdot\log(\frac{d}{\epsilon}))$, the algorithm `PowerMethod`$(\mathbf{A}, k)$ outputs a vector u such that*

$$\langle v_1, u\rangle^2 \geq 1-\epsilon \text{ and } u^\top \mathbf{A} u \geq (1-\epsilon)\lambda_1$$

with high probability in d.

Proof. We write $u = \sum_{i\in[d]} \alpha_i v_i$. Then, we have

$$\langle v_1, u\rangle^2 = \left\langle v_1, \frac{\sum_{i\in[d]}\alpha_i\lambda_i^k v_i}{\sqrt{\sum_{i\in[d]}\alpha_i^2\lambda_i^{2k}}}\right\rangle^2 = \frac{\alpha_1^2}{\alpha_1^2 + \sum_{j\neq 1}\alpha_j^2\left(\frac{\lambda_j}{\lambda_1}\right)^{2k}}$$
$$\geq \frac{\alpha_1^2/\|x\|_2^2}{\alpha_1^2/\|x\|_2^2 + e^{-2kg}}$$

where we used that $\frac{\lambda_2}{\lambda_1} = 1 - g \leq e^{-g}$. With high probability in $1 - \frac{1}{d^c}$, we have $\|x\|_2^2 \leq 2d$ and $|\alpha_1| \geq \frac{1}{d^{O(c)}}$. In this case, we have $\alpha_1^2/\|x\|_2^2 \geq 1/d^{O(c)}$.

Using $k = \Omega\left(\frac{1}{g}\log\left(\frac{d}{\epsilon}\right)\right)$, we have $\langle v_1, u\rangle^2 \geq 1-\epsilon$. Furthermore, this implies that

$$u^\top \mathbf{A} u = u^\top\left(\sum_{i\in[d]}\lambda_i v_i v_i^\top\right)u \geq \lambda_1\langle v_1, u\rangle^2 \geq (1-\epsilon)\lambda_1.$$

□

C.4 Noisy 1-D Convex Optimization

Here we show how to minimize an one dimensional convex function giving a noisy oracle for evaluating the function. While this could possibly be done using general results on convex optimization using a membership oracle, the proof in one dimension is much simpler and we include it here for completeness .

Algorithm 8: `OneDimMinimizer`$(\ell, u, \epsilon, g, L)$

Input: Interval $[\ell, u] \subseteq \mathbb{R}$, target additive error ϵ, noisy additive evaluation oracle g, Lipschitz bound L
Let $x^{(0)} = \ell$, $y_\ell^{(0)} = \ell$, $y_u^{(0)} = u$
for $i = 1, ..., \left\lceil \log_{3/2}(\frac{L(u-\ell)}{\epsilon})\right\rceil$ **do**
  Let $z_\ell^{(i)} = \frac{2y_\ell^{(i-1)} + y_u^{(i-1)}}{3}$, $z_u^{(i)} = \frac{y_\ell^{(i-1)} + 2y_u^{(i-1)}}{3}$...
  if $g(z_\ell^{(i)}) \leq g(z_u^{(i)})$ **then**
    Let $(y_\ell^{(i)}, y_u^{(i)}) = (y_\ell^{(i-1)}, z_u^{(i)})$.
    Let $x^{(i)} = z_\ell^{(i)}$ if $g(z_\ell^{(i)}) \leq g(x^{(i-1)})$ and $x^{(i-1)}$ otherwise.
  if $g(z_\ell^{(i)}) > g(z_u^{(i)})$ **then**
    Let $(y_\ell^{(i)}, y_u^{(i)}) = (z_\ell^{(i)}, y_u^{(i-1)})$.
    Let $x^{(i)} = z_u^{(i)}$ if $g(z_u^{(i)}) \leq g(x^{(i-1)})$ and $x^{(i-1)}$ otherwise.
end
Output: x

Lemma C.8. *Given a L-Lipschitz convex function defined in $[\ell, u]$ interval. Suppose that we have an oracle g such that $|g(y) - f(y)| \leq \epsilon$ for all y. In $O(\log(\frac{L(u-\ell)}{\epsilon}))$ time and and with $O(\log(\frac{L(u-\ell)}{\epsilon}))$ calls to g, `OneDimMinimizer`$(\ell, u, \epsilon, g, L)$ outputs a point x such that*

$$f(x) - \min_{y\in[\ell,u]} f(y) \leq 4\epsilon.$$

Proof. First, note that for any y, y', if $f(y) < f(y') - 2\epsilon$,$g(y) < g(y')$. This directly follows from the error guarantee on g.

Now, the output of the algorithm, x, is simply the point queried by the algorithm (i.e. ℓ and the z_ℓ^i and z_u^i) with the smallest value of g. This implies that $f(x)$ is within 2ϵ of the minimum value of f among the points queried. It thus suffices to show that the algorithm queries some point within 2ϵ of optimal.

To do this, we break into two cases. One is where the intervals $[y_\ell^i, y_u^i]$ all contain an optimum of f. In this case, the final interval contains an optimum, and is of size at most $\frac{\epsilon}{L}$. Thus, by the Lipschitz property, all points in the interval are within $\epsilon \leq 2\epsilon$ of optimal, and at least one endpoint of the interval must have been queried by the algorithm.

For the other case, we consider the last i for which this interval does contain an optimum of f. This means that $g(z_\ell^{(i)}) \leq g(z_u^{(i)})$ while the optimum x^* is to the right of $z_u^{(i)}$, or the symmetric case with the optimum to the left of $g(z_\ell^{(i)})$. Without loss of generality, we will assume the former. We then have $z_\ell^{(i)} \leq z_u^{(i)} \leq x^*$, while $x^* - z_u^{(i)} \leq z_u^{(i)} - z_\ell^{(i)}$. By the convexity of f, we therefore have that $f(z_u^{(i)}) - f(x^*) \leq f(z_\ell^{(i)}) - f(z_u^{(i)})$. But $f(z_\ell^{(i)}) - f(z_u^{(i)}) \leq 2\epsilon$ since $g(z_\ell^{(i)}) \leq g(z_u^{(i)})$. Thus, $f(z_u^{(i)}) - f(x^*) \leq 2\epsilon$, and $z_u^{(i)}$ is queried by the algorithm, as desired. □

D. WEIGHTED GEOMETRIC MEDIAN

In this section we show how to extend our results to the *weighted geometric median* problem, also known as the Weber problem. Given a set of n points in d dimensions, $a^{(1)}, \ldots, a^{(n)} \in \mathbb{R}^d$, with corresponding non-negative weights we denote as and $w^{(1)}, \ldots, w^{(n)} \in \mathbb{R}_{>0}$, find a point $x^* \in \mathbb{R}^d$ that minimizes the weighted sum of Euclidean distances to them:

$$x^* \in \arg\min_{x \in \mathbb{R}^d} f(x) \quad \text{where} \quad f(x) \stackrel{\text{def}}{=} \sum_{i \in [n]} w^{(i)} \|x - a^{(i)}\|_2.$$

As in the unweighted problem, our goal is to compute a $(1+\epsilon)$-approximate solution, i.e. $x \in \mathbb{R}^d$ with $f(x) \leq (1+\epsilon) f(x^*)$.

First, we show that it suffices to consider the case where the weights are integers with bounded sum (Lemma D.1). Then, we show that such an instance of the weighted geometric median problem can be solved using the algorithms developed for the unweighted problem.

LEMMA D.1. *Given points $a^{(1)}, a^{(2)}, \ldots, a^{(n)} \in \mathbb{R}^d$, non-negative weights $w^{(1)}, w^{(2)}, \ldots, w^{(n)} \in \mathbb{R}_{>0}$, and $\epsilon \in (0,1)$, we can compute in linear time weights $w_1^{(1)}, w_1^{(2)}, \ldots, w_1^{(n)}$ such that:*

- *Any $(1+\epsilon/5)$-approximate weighted geometric median of $a^{(1)}, \ldots, a^{(n)}$ with the weights $w_1^{(1)}, \ldots, w_1^{(n)}$ is also a $(1+\epsilon)$-approximate weighted geometric median of $a^{(1)}, \ldots, a^{(n)}$ with the weights $w^{(1)}, \ldots, w^{(n)}$, and*
- *$w_1^{(1)}, \ldots, w_1^{(n)}$ are nonnegative integers and $\sum_{i=1}^n w_1^{(i)} \leq 5n\epsilon^{-1}$.*

PROOF. Let

$$f(x) = \sum_{i \in [n]} w^{(i)} \|a^{(i)} - x\|$$

and $W = \sum_{i \in [n]} w^{(i)}$. Furthermore, let $\epsilon' = \epsilon/5$ and for each $i \in [n]$, define

$$\begin{aligned} w_0^{(i)} &= \frac{n}{\epsilon' W} w^{(i)} \\ w_1^{(i)} &= \left\lfloor w_0^{(i)} \right\rfloor \\ w_2^{(i)} &= w_0^{(i)} - w_1^{(i)} \end{aligned}$$

We also define $f_0, f_1, f_2, W_0, W_1, W_2$ analogously to f and W.

Now, assume $f_1(x) \leq (1+\epsilon') f_1(x^*)$, where x^* is the minimizer of f and f_0. Then:

$$\begin{aligned} f_0(x) &= f_1(x) + f_2(x) \\ &\leq f_1(x) + f_2(x^*) + W_2 \|x - x^*\|_2 \end{aligned}$$

and

$$\begin{aligned} W_2 \|x - x^*\|_2 &= \frac{W_2}{W_1} \sum_{i \in [n]} w_1^{(i)} \|x - x^*\|_2 \\ &\leq \frac{W_2}{W_1} \sum_{i \in [n]} w_1^{(i)} \left(\|x - a^{(i)}\|_2 + \|x^* - a^{(i)}\|_2 \right) \\ &\leq \frac{W_2}{W_1} \left(f_1(x) + f_1(x^*) \right) \end{aligned}$$

Now since $W_0 = \frac{n}{\epsilon'}$ and $W_1 \geq W_0 - n$ we have

$$\begin{aligned} \frac{W_2}{W_1} = \frac{W_0 - W_1}{W_1} = \frac{W_0}{W_1} - 1 &\leq \frac{W_0}{W_0 - n} - \frac{W_0 - n}{W_0 - n} \\ &= \frac{n}{\frac{n}{\epsilon'} - n} = \frac{\epsilon'}{1 - \epsilon'} \end{aligned}$$

Combining these yields that

$$\begin{aligned} f_0(x) &\leq f_1(x) + f_2(x^*) + \frac{\epsilon'}{1-\epsilon'} (f_1(x) + f_1(x^*)) \\ &\leq \left(1 + \frac{\epsilon'}{1-\epsilon'}\right) (1+\epsilon') f_1(x^*) \\ &\quad + \frac{\epsilon'}{1-\epsilon'} f_1(x^*) + f_2(x^*) \\ &\leq (1 + 5\epsilon') f_0(x^*) \\ &= (1+\epsilon) f_0(x^*) \end{aligned}$$

□

We now proceed to show the main result of this section.

LEMMA D.2. *A $(1+\epsilon)$-approximate weighted geometric median of n points in $\mathbb{R}^d$ can be computed in $O(nd \log^3 \epsilon^{-1})$ time.*

PROOF. By applying Lemma D.1, we can assume that the weights are integer and their sum does not exceed $n\epsilon^{-1}$. Note that computing the weighted geometric median with such weights is equivalent to computing an unweighted geometric median of $O(n\epsilon^{-1})$ points (where each point of the original input is repeated with the appropriate multiplicity). We now show how to simulate the behavior of our unweighted geometric median algorithms on such a set of points without computing it explicitly.

If $\epsilon > n^{-1/2}$, we will apply `ApproximateMedian`(ϵ), achieving a runtime of $O(d\epsilon^{-2}) = O(nd)$. It is only necessary to check that we can implement weighted sampling from our points with $O(n)$ preprocessing and $O(1)$ time per sample. This is achieved by the alias method [15].

Now assume $\epsilon < n^{-1/2}$. We will employ the algorithm `AccurateMedian`(ϵ). Note that initialization using the algorithm `ApproximateMedian`(2) can be performed as described in the previous paragraph. It remains to check that we can implement `LineSearch` and `ApproxMinEig` on the implicitly represented multiset of $O(n\epsilon^{-1})$ points. It is enough to observe only n of the points are distinct, and all computations performed by these subroutines are identical for identical points. The total runtime will thus be $O(nd \log^3(n/\epsilon^2)) = O(nd \log^3 \epsilon^{-1})$. □