On Khachiyan's Algorithm for the Computation of Minimum Volume Enclosing Ellipsoids

Michael J. Todd[*] E. Alper Yıldırım[†]

September 30, 2005

Dedicated to the memory of Leo Khachiyan

Abstract

Given $\mathcal{A} := \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ whose affine hull is $\mathbb{R}^d$, we study the problems of computing an approximate rounding of the convex hull of $\mathcal{A}$ and an approximation to the minimum volume enclosing ellipsoid of $\mathcal{A}$. In the case of centrally symmetric sets, we first establish that Khachiyan's barycentric coordinate descent (BCD) method is exactly the polar of the deepest cut ellipsoid method using two-sided symmetric cuts. This observation gives further insight into the efficient implementation of the BCD method. We then propose a new algorithm which computes an approximate rounding of the convex hull of $\mathcal{A}$, and which can also be used to compute an approximation to the minimum volume enclosing ellipsoid of $\mathcal{A}$. Our algorithm is a modification of the algorithm of Kumar and Yıldırım, which combines Khachiyan's BCD method with a simple initialization scheme to achieve a slightly improved polynomial complexity result, and which returns a small "core set." We establish that our algorithm computes an approximate solution to the dual optimization formulation of the minimum volume enclosing ellipsoid problem that satisfies a more complete set of approximate optimality conditions than either of the two previous algorithms. Furthermore, this added benefit is achieved without any increase in the improved asymptotical complexity bound of the algorithm of Kumar and Yıldırım or any increase in the bound on the size of the computed core set. In addition, the "dropping idea" used in our algorithm has the potential of computing smaller core sets in practice. We also discuss several possible variants of this dropping technique.

Keywords: Löwner ellipsoid, core sets, rounding of polytopes, ellipsoid method, approximation algorithms.

[*]School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 14853, USA. This author was supported in part by NSF through grant DMS-0513337 and ONR through grant N00014-02-1-0057. e-mail: `miketodd@cs.cornell.edu`

[†]Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey. (On leave from the Department of Applied Mathematics and Statistics at Stony Brook University, Stony Brook, NY, 11794, USA.) This author was supported in part by NSF through CAREER grant DMI-0237415. e-mail: `yildirim@bilkent.edu.tr`

1 Introduction

Let $\mathcal{A} := \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ be a finite set of vectors whose affine hull is $\mathbb{R}^d$. In this paper, we are concerned with the problem of computing an approximate "rounding" of the convex hull of $\mathcal{A}$ as well as the problem of computing the minimum volume enclosing ellipsoid (MVEE) of $\mathcal{A}$, which we shall denote by MVEE($\mathcal{A}$), also known as the Löwner or Löwner-John ellipsoid of $\mathcal{A}$.

Minimum volume enclosing ellipsoids play an important role in several diverse applications such as optimal design (Silvey and Titterington [27, 29]), computational geometry (Welzl [34], Chazelle and Matoušek [10], and Barequet and Har-Peled [4]), convex optimization (Grötschel et al. [16] and Nesterov [24]), computer graphics (Eberly [11] and Bouville [5]), pattern recognition (Glineur [14]), and statistics (Silverman and Titterington [26]).

The minimum volume ellipsoid satisfies (John [17])

$$\frac{1}{d}\,\mathrm{MVEE}(\mathcal{A}) \subseteq \mathrm{conv}(\mathcal{A}) \subseteq \mathrm{MVEE}(\mathcal{S}), \tag{1}$$

where conv($\mathcal{A}$) denotes the convex hull of $\mathcal{A}$ and the ellipsoid on the left-hand side is obtained by scaling MVEE($\mathcal{A}$) around its center by a factor of $1/d$. Furthermore, if $\mathcal{A}$ is centrally symmetric (i.e., if $\mathcal{A} = -\mathcal{A}$), the factor on the left-hand side can be improved to $1/\sqrt{d}$. Therefore, MVEE($\mathcal{A}$) provides a rounding of the full-dimensional polytope conv($\mathcal{A}$).

Given $\epsilon > 0$, an ellipsoid $\mathcal{E} \subset \mathbb{R}^d$ is said to be a $(1+\epsilon)d$-rounding of conv($\mathcal{A}$) if

$$\frac{1}{(1+\epsilon)d}\,\mathcal{E} \subseteq \mathrm{conv}(\mathcal{A}) \subseteq \mathcal{E}. \tag{2}$$

In the case of centrally symmetric point sets, we replace the factor on the left-hand side of (2) by $1/\sqrt{(1+\epsilon)d}$.

Similarly, for $\eta > 0$, we say that an ellipsoid $\mathcal{E} \subset \mathbb{R}^d$ is a $(1+\eta)$-approximation of MVEE($\mathcal{A}$) if

$$\mathrm{conv}(\mathcal{A}) \subseteq \mathcal{E}, \quad \mathrm{Vol}(\mathcal{E}) \leq (1+\eta)\,\mathrm{Vol}(\mathrm{MVEE}(\mathcal{A})), \tag{3}$$

where Vol($\mathcal{E}$) denotes the volume of the ellipsoid $\mathcal{E}$.

Several algorithms have been developed for the MVEE problem. These algorithms can be categorized as first-order algorithms [29, 30, 26, 18, 21], second-order interior-point algorithms [25, 28], or a combination of the two [18]. For small dimensions d, the MVEE problem can be solved in $O(d^{O(d)}m)$ arithmetic operations using randomized [22, 34, 1] or deterministic [10] algorithms. A fast implementation is also available in the CGAL library[1] for solving the problem in two dimensions [12]. Khachiyan and Todd [19] established a linear-time reduction of the MVEE problem to the problem of computing a maximum volume inscribed ellipsoid (MVIE) in a polytope described by a finite number of inequalities. Therefore, the MVEE problem can also be solved using the algorithms developed for the MVIE problem [19, 23, 35, 3, 36]. Since the MVEE problem can be formulated as a maximum

[1]http://www.cgal.org

determinant problem, the more general algorithms of Vandenberghe et al. [33] and Toh [32] can also be applied.

Khachiyan [18] proposed an algorithm to compute a $(1+\epsilon)d$-rounding of $\text{conv}(\mathcal{A})$ in polynomial time for fixed $\epsilon > 0$; it can also compute a $(1+\eta)$-approximation of $\text{MVEE}(\mathcal{A})$ in polynomial time for fixed $\eta > 0$. Khachiyan's algorithm can be viewed as a barycentric coordinate descent (BCD) method [18] or as a Frank-Wolfe algorithm or sequential linear programming algorithm [28, 21] for the dual optimization formulation of the MVEE problem. Khachiyan's algorithm computes a $(1+\eta)$-approximation to $\text{MVEE}(\mathcal{A})$ in

$$O\left(md^2\left([(1+\eta)^{2/(d+1)}-1]^{-1}+\log d+\log\log m\right)\right) \tag{4}$$

operations for $\eta > 0$. (Note that $[(1+\eta)^{2/(d+1)}-1]^{-1} = O(d/\eta)$ for $\eta \in (0,1]$.)

More recently, Kumar and Yıldırım [21] proposed a modification of Khachiyan's algorithm (henceforth the KY algorithm) using a simple initialization scheme, which can compute a $(1+\eta)$-approximation to $\text{MVEE}(\mathcal{A})$ in

$$O\left(md^2\left([(1+\eta)^{2/(d+1)}-1]^{-1}+\log d\right)\right) \tag{5}$$

operations, which is a slight improvement over (4). In addition, the algorithm of [21] computes an ellipsoid $\mathcal{E} \subset \mathbb{R}^d$ such that $\mathcal{A} \subseteq \mathcal{E}$ and a subset $\mathcal{X} \subseteq \mathcal{A}$ with the property that

$$\text{Vol}(\text{MVEE}(\mathcal{X})) \leq \text{Vol}(\text{MVEE}(\mathcal{A})) \leq \text{Vol}(\mathcal{E}) \leq (1+\eta)\text{Vol}(\text{MVEE}(\mathcal{X})) \leq (1+\eta)\text{Vol}(\text{MVEE})(\mathcal{A}),$$

which implies that $\mathcal{E}$ is simultaneously a $(1+\eta)$-approximation to $\text{MVEE}(\mathcal{X})$ and to $\text{MVEE}(\mathcal{A})$. Such a set $\mathcal{X}$ is called an "η-core set" (or a core set) for $\mathcal{A}$ to signify that $\text{conv}(\mathcal{X})$ provides a good approximation of $\text{conv}(\mathcal{A})$. Furthermore, $\mathcal{X}$ satisfies

$$|\mathcal{X}| = O\left(d[(1+\eta)^{2/(d+1)}-1]^{-1}+d\log d\right), \tag{6}$$

which is independent of m, the number of points in $\mathcal{A}$. (Note that John [17] shows that a 0-core set of cardinality at most $(d+1)(d+2)/2$ exists.) "Small" core set results have previously been established for several geometric optimization problems [20, 7, 6, 9, 2] and play an important role in developing efficient and practical algorithms for various large-scale problems in moderate dimensions.

Our contributions in this paper are twofold. In the case of centrally symmetric point sets, we establish that Khachiyan's BCD method is exactly the polar of the deepest cut ellipsoid method (using two-sided symmetric cuts), but with a much improved analysis. Secondly, for arbitrary point sets, we propose a modification of the algorithm of [21], which computes an approximate solution to the more complete optimality conditions than either of the algorithms in [18] or in [21]. Furthermore, we establish that our modification does not lead to any increase in the asymptotic complexity of the algorithm of [21] given by (5). Finally, the KY algorithm of [21] starts with a carefully selected, small core set and expands it gradually. In contrast, our algorithm allows for dropping points in the working core set, which has the potential to compute a smaller core set than that given by (6) (but certainly no bigger asymptotically).

The paper is organized as follows. We define notation in the remainder of this section. In Section 2, we review formulations of the MVEE problem as an optimization problem and we describe various approximate optimality conditions. Section 3 discusses an interpretation of Khachiyan's algorithm as the polar of the deepest symmetric cut ellipsoid algorithm. Then in Section 4, we describe and analyze our modification of this and the KY algorithm. We conclude the paper in Section 5.

1.1 Notation

Vectors are denoted by lower-case Roman letters. For a vector p, p_i denotes its ith component, and P the diagonal matrix whose diagonal entries are given by these components. Inequalities on vectors apply to each component. We reserve e for the vector of ones of appropriate dimension, which will be clear from the context, and e^j for the jth unit vector. Upper-case Roman letters are reserved for matrices. For a finite set of vectors $\mathcal{V}$, span($\mathcal{V}$) denotes the linear subspace spanned by $\mathcal{V}$. Functions and operators are denoted by upper-case Greek letters. Scalars except for m, d, and n are represented by lower-case Greek letters unless they represent components of a vector or a sequence of scalars, vectors or matrices. We reserve i, j, and k for indexing purposes. Upper-case script letters are used for all other objects such as sets, polytopes, and ellipsoids.

2 Preliminaries and Formulations

A (full-dimensional) ellipsoid $\mathcal{E}_{Q,c}$ in $\mathbb{R}^d$ is specified by a $d \times d$ symmetric positive definite matrix Q and a center $c \in \mathbb{R}^d$ and admits a representation given by

$$\mathcal{E}_{Q,c} = \{x \in \mathbb{R}^d : (x - c)^T Q (x - c) \leq 1\}. \tag{7}$$

The volume of the ellipsoid $\mathcal{E}_{Q,c}$, denoted by Vol($\mathcal{E}_{Q,c}$), is given by $\text{Vol}(\mathcal{E}_{Q,c}) = \rho \det Q^{-\frac{1}{2}}$, where ρ is the volume of the unit ball in $\mathbb{R}^d$ [16]. Similarly, we define the scaled volume by

$$\text{vol}(\mathcal{E}_{Q,c}) := \det Q^{-\frac{1}{2}}. \tag{8}$$

Let $\mathcal{A} := \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ be a finite set of vectors whose affine hull is $\mathbb{R}^d$. If $\mathcal{A}$ is not centrally symmetric, we define a "lifting" of $\mathcal{A}$ to $\mathbb{R}^n$, where $n := d + 1$, by

$$\mathcal{A}' := \{\pm q^1, \ldots, \pm q^m\}, \quad \text{where} \quad q^i := \begin{bmatrix} a^i \\ 1 \end{bmatrix}, \quad i = 1, \ldots, m, \tag{9}$$

which is centrally symmetric. It turns out that MVEE($\mathcal{A}$) and MVEE($\mathcal{A}'$) are closely related [19, 25]. More specifically,

$$\text{MVEE}(\mathcal{A}) \times \{1\} = \text{MVEE}(\mathcal{A}') \cap \Pi, \tag{10}$$

where

$$\Pi := \{x \in \mathbb{R}^n : x_n = 1\}. \tag{11}$$

In addition, for any $\eta > 0$, if $\mathcal{E}' \subset \mathbb{R}^n$ is a $(1+\eta)$-approximation of MVEE($\mathcal{A}'$), then $\mathcal{E} := \mathcal{E}' \cap \Pi$ is a $(1+\eta)$-approximation of MVEE($\mathcal{A}$) [19]. Henceforth, we assume that $\mathcal{A}$ is not centrally symmetric; clearly, if it is, our algorithms can be simplified by omitting this lifting step.

Since $\mathcal{A}'$ is centrally symmetric, MVEE($\mathcal{A}'$) is centered at the origin. Therefore, the problem of computing MVEE($\mathcal{A}'$) can be formulated as the following convex optimization problem:

$$\begin{array}{llll} (\mathbf{P}(\mathcal{A}')) & \min_M & -\log\det M & \\ & \text{s.t.} & (q^i)^T M\, q^i & \leq 1, \quad i = 1, \dots, m, \\ & & M \in \mathbb{R}^{n \times n} & \text{is symmetric and positive definite,} \end{array}$$

where $M \in \mathbb{R}^{n \times n}$ is the decision variable.

The Lagrangian dual of $(\mathbf{P}(\mathcal{A}'))$ is equivalent to

$$\begin{array}{lll} (\mathbf{D}(\mathcal{A}')) & \max_p & \Phi(p) := \log\det \Lambda(p) \\ & \text{s.t.} & e^T p = 1, \\ & & p \geq 0, \end{array}$$

where $p \in \mathbb{R}^m$ is the decision variable and $\Lambda : \mathbb{R}^m \to \mathbb{R}^{n \times n}$ is the linear operator given by

$$\Lambda(p) := \sum_{i=1}^m p_i\, q^i (q^i)^T. \tag{13}$$

The necessary and sufficient optimality conditions for p^* to solve $(\mathbf{D}(\mathcal{A}'))$ are given by

$$w_i(p^*) + s_i^* = \lambda^*, \quad i = 1, \dots, m, \tag{14a}$$

$$e^T p^* = 1, \tag{14b}$$

$$p_i^* s_i^* = 0, \quad i = 1, \dots, m, \tag{14c}$$

together with $p^* \geq 0$ and $s^* \geq 0$, where

$$w_i(p) := (q^i)^T \Lambda(p)^{-1} q^i, \quad i = 1, \dots, m. \tag{15}$$

For any feasible solution $p \in \mathbb{R}^m$ of $(\mathbf{D}(\mathcal{A}'))$ with $\Phi(p) > -\infty$, we have [18, Lemma 1]

$$\sum_{i=1}^m p_i\, w_i(p) = n. \tag{16}$$

Therefore, multiplying both sides of (14a) by p_i^* and summing up for $i = 1, \dots, m$, we obtain $\lambda^* = n$ by (14b) and (14c). It follows then that the optimality conditions of $(\mathbf{D}(\mathcal{A}'))$ can be equivalently rewritten as

$$w_i(p^*) \leq n, \quad i = 1, \dots, m, \tag{17a}$$

$$e^T p^* = 1, \tag{17b}$$

together with $p^* \geq 0$. By (17) and (16),

$$p_i^* > 0 \text{ implies } w_i(p^*) = n, \; i = 1, \ldots, m, \tag{18}$$

which are simply the complementary slackness conditions (14c).

Khachiyan's algorithm [18] is driven by computing a feasible solution $\tilde{p}$ of $(\mathbf{D}(\mathcal{A}'))$ that satisfies the so-called ϵ-relaxed optimality conditions defined by

$$w_i(\tilde{p}) \leq (1+\epsilon)n, \quad i = 1, \ldots, m. \tag{19}$$

For such a solution $\tilde{p}$, let $j \in \{1, \ldots, m\}$ be such that $\tilde{p}_j > 0$. By (16),

$$\begin{aligned} w_j(\tilde{p}) &= \frac{1}{\tilde{p}_j}\left(n - \sum_{i=1, i \neq j}^{m} \tilde{p}_i w_i(\tilde{p})\right), \\ &\geq \left(n[1-(1+\epsilon)(1-\tilde{p}_j)]\right)/\tilde{p}_j, \\ &= n\left(1 + \epsilon - \epsilon/\tilde{p}_j\right), \end{aligned}$$

where we used (19) and the feasibility of $\tilde{p}$ to derive the inequality. Therefore, such a solution $\tilde{p}$ satisfies a very weak approximate form of the complementary slackness conditions (18).

In view of this observation, we define a more complete set of approximate optimality conditions aimed towards a stronger approximation to the complementary slackness conditions (18). Given $\epsilon \in (0,1)$, we say that a feasible solution $\hat{p}$ satisfies the ϵ-approximate optimality conditions if

$$w_i(\hat{p}) \quad \leq \quad (1+\epsilon)n, \quad i = 1, \ldots, m, \tag{20a}$$
$$\hat{p}_i > 0 \quad \text{implies} \quad w_i(\hat{p}) \geq (1-\epsilon)n, \quad i = 1, \ldots, m. \tag{20b}$$

Note that these conditions imply that $(1-\epsilon)n \leq w_i(\hat{p}) \leq (1+\epsilon)n$ if $\hat{p}_i > 0, \; i = 1, \ldots, m$, which is a better approximation of the complementary slackness conditions (18) than the ϵ-relaxed optimality conditions (19) provide.

Khachiyan's algorithm [18] starts with a feasible solution $\bar{p} > 0$ of $(\mathbf{D}(\mathcal{A}'))$ and improves upon the objective function value by increasing only one component of $\bar{p}$ at each iteration and then rescaling to regain feasibility. On the other hand, the KY algorithm [21] uses a simple initialization scheme to compute a feasible solution $\dot{p}$ of $(\mathbf{D}(\mathcal{A}'))$ with only $\min\{2d, m\}$ positive components and then uses the same improvement idea. Therefore, both of these algorithms can only add to the number of positive components of p at each iteration. In contrast, while our algorithm starts off with the same initial feasible solution $\dot{p}$ of $(\mathbf{D}(\mathcal{A}'))$, we will allow reductions in some positive components of p. Therefore, our algorithm can potentially compute an approximate solution with a smaller number of positive components than an approximate solution computed by either of the previous two algorithms, which, in turn, will result in a smaller core set. In addition, the asymptotic complexity of our algorithm remains the same as the improved complexity result (5) of [21].

We close this section by relating the optimal solution p^* of $(\mathbf{D}(\mathcal{A}'))$ to MVEE$(\mathcal{A})$ (see, e.g., [21, Lemma 2.1]). Let $A \in \mathbb{R}^{d\times m}$ be the matrix whose ith column is given by $a^i,\ i = 1, \dots, m$. Then MVEE$(\mathcal{A}) = \mathcal{E}_{Q^*,c^*} := \{x \in \mathbb{R}^d : (x - c^*)^T Q^* (x - c^*) \le 1\}$, where

$$Q^* := \frac{1}{d}\left(AP^*A^T - Ap^*(Ap^*)^T\right)^{-1}, \quad c^* := Ap^*. \tag{21}$$

Furthermore,

$$\log \mathrm{vol}(\mathrm{MVEE}(\mathcal{A})) = \frac{d}{2}\log d + \frac{1}{2}\log\det \Lambda(p^*). \tag{22}$$

3 A New Interpretation of Khachiyan's Algorithm

In this section, we establish that Khachiyan's algorithm [18] is exactly the polar of the deepest cut ellipsoid method (using two-sided symmetric cuts), but with a much improved analysis.

We describe Khachiyan's algorithm below.

Algorithm 3.1 Khachiyan's algorithm to compute a feasible solution of $(\mathbf{D}(\mathcal{A}'))$ satisfying (19).

```
Require: Input set of points $\mathcal{A} = \{a^1, \dots, a^m\} \subset \mathbb{R}^d, \epsilon > 0$.
1: $k \leftarrow 0$, $n \leftarrow d+1$, $p^0 \leftarrow (1/m)e$, and $q^i \leftarrow ((a^i)^T, 1)^T, i = 1, \dots, m$.
2: While $p^k$ does not satisfy (19), do
3: loop
4:     $j \leftarrow \arg\max_{i=1,\dots,m} (q^i)^T \Lambda(p^k)^{-1} q^i$, $\kappa \leftarrow (q^j)^T \Lambda(p^k)^{-1} q^j$;
5:     $\beta \leftarrow \frac{\kappa - n}{n(\kappa - 1)}$;
6:     $p^{k+1} \leftarrow (1-\beta)p^k + \beta e^j$, $k \leftarrow k+1$.
7: end loop
8: Output $p^k$.
```

Khachiyan's algorithm seeks a minimum volume ellipsoid containing $\mathcal{Q} := \mathrm{conv}\{\pm q^1, \cdots, \pm q^m\}$, but, since it is a dual algorithm, it constructs a sequence of ellipsoids

$$\mathcal{E}_k := \{y \in \mathbb{R}^n : y^T \Lambda(p^k)^{-1} y \le 1\}$$

satisfying $\mathcal{E}_k \subseteq \mathcal{Q}$, and stops when $\mathcal{Q} \subseteq \sqrt{(1+\epsilon)n}\,\mathcal{E}_k$. Thus the polar ellipsoids

$$\mathcal{E}_k^\circ = \{z \in \mathbb{R}^n : z^T \Lambda(p^k) z \le 1\} \tag{23}$$

all contain the polar polytope

$$\mathcal{Q}^\circ = \{z \in \mathbb{R}^n : -1 \le (q^i)^T z \le 1,\ i = 1, \cdots, m\},$$

and the algorithm stops when $(1/\sqrt{(1+\epsilon)n})\mathcal{E}_k^\circ$ is contained in $\mathcal{Q}^\circ$.

So suppose that, at a particular iteration, we have the ellipsoid $\mathcal{E}_k^\circ$, and that $(1/\sqrt{(1+\epsilon)n})\mathcal{E}_k^\circ$ is not contained in $\mathcal{Q}^\circ$. That means that one of the pairs of hyperplanes $(q^i)^T z = \pm 1$ of $\mathcal{Q}^\circ$ intersects $\mathcal{E}_k^\circ$ "close to the origin." So the condition $(q^j)^T \Lambda(p^k)^{-1} q^j =: \kappa > (1+\epsilon)n$ in Khachiyan's algorithm corresponds to

$$\mathcal{Q}^\circ \subseteq \{z \in \mathcal{E}_k^\circ : -\gamma((q^j)^T \Lambda(p^k)^{-1} q^j)^{1/2} \le (q^j)^T z \le \gamma((q^j)^T \Lambda(p^k)^{-1} q^j)^{1/2}\}$$

for $\gamma := 1/\sqrt{\kappa} < 1/\sqrt{(1+\epsilon)n}$. (We use γ here for β in [31], to avoid confusion with Khachiyan's β.) This is exactly the set-up of the deepest two-sided symmetric cut ellipsoid method, which chooses as the next ellipsoid the smallest volume ellipsoid containing the right-hand side above. Further, in the version of Burrell and Todd [8], the initial ellipsoid is also of the form in (23) for a suitable p.

Let us examine the next ellipsoid constructed in this ellipsoid method. According to Theorem 2 of [31], this has the form $\{z \in \mathbb{R}^n : z^T B_+^{-1} z \le 1\}$, with

$$B_+ := \delta \left(B - \sigma \frac{(Bq)(Bq)^T}{q^T B q} \right), \tag{24}$$

where $B := \Lambda(p^k)^{-1}$, $q := q^j$, and

$$\delta := \frac{n(1-\gamma^2)}{n-1}, \quad \sigma := \frac{1-n\gamma^2}{1-\gamma^2}.$$

Then, using the rank-one modification formula, we obtain

$$\begin{aligned} B_+^{-1} &= \delta^{-1}\left(B^{-1} + \frac{\sigma}{(1-\sigma) q^T B q} q q^T \right) \\ &= \delta^{-1}\left(B^{-1} + \frac{\sigma\gamma^2}{(1-\sigma)} q q^T \right) \\ &= \delta^{-1}(1+\mu)\left[\left(1 - \frac{\mu}{1+\mu}\right) B^{-1} + \frac{\mu}{1+\mu} q q^T \right], \end{aligned}$$

where

$$\mu := \frac{\sigma\gamma^2}{(1-\sigma)}.$$

Now substituting in the value for σ, we find $\mu = (1-n\gamma^2)/(n-1)$, so that $1+\mu = n(1-\gamma^2)/(n-1)$, and

$$\frac{\mu}{1+\mu} = \frac{1-n\gamma^2}{n(1-\gamma^2)} = \frac{\gamma^{-2}-n}{n(\gamma^{-2}-1)} = \frac{\kappa-n}{n(\kappa-1)} = \beta,$$

using $\gamma := 1/\sqrt{\kappa}$. Next, substituting in the value for δ, we get $\delta^{-1}(1+\mu) = 1$, so that

$$B_+^{-1} = (1-\beta)B^{-1} + \beta q q^T = (1-\beta)\Lambda(p^k) + \beta q q^T = \Lambda(p^{k+1}).$$

This demonstrates the equivalence of the two methods.

Above, we claimed that Khachiyan provides a much improved analysis of his method. Indeed, he showed (Lemma 3 of [18]) that, at every iteration, the (natural) logarithm of the volume of $\mathcal{E}_k$ increases by $(1/2)(\log(1+\epsilon_k) - \epsilon_k/(1+\epsilon_k))$, where $\max_{i=1,\ldots,m}(q^i)^T\Lambda(p^k)^{-1}q^i = (1+\epsilon_k)n$. He then uses this in his Lemma 4 to provide a bound on the total number of iterations, by bounding the number required to reduce ϵ_k to 1, then to 1/2, etc. By contrast, analyses of the ellipsoid method just look at the worst case volume reduction: in this case, one would bound the number of iterations assuming that the logarithm of the volume only increased by $(1/2)(\log(1+\epsilon) - \epsilon/(1+\epsilon))$ at each iteration, where ϵ is the final tolerance.

Remark: Let us briefly discuss implementation of Khachiyan's algorithm (and our variants). At each iteration, we need access to the inverse of $\Lambda(p^k)$, or to some factorization of it, and to the quantities $(q^i)^T\Lambda(p^k)^{-1}q^i$ for each i. At each iteration, $\Lambda(p^k)$ is updated by adding a rank-one symmetric matrix to it, and then scaling by a positive number. In some applications, the vectors q^i and hence possibly the matrix $\Lambda(p^k)$ will be sparse. We therefore recommend maintaining a Cholesky factorization LDL^T of $\Lambda(p^k)$, where L is a lower triangular matrix with unit diagonal and D is diagonal with positive diagonal entries. We also maintain values $\kappa_i = w_i(p^k) = (q^i)^T\Lambda(p^k)^{-1}q^i$ for each i. From these values we can determine κ and j. We then compute $\hat{q}^j := L^{-1}q^j$ and from this a more accurate value $(\hat{q}^j)^T D^{-1}\hat{q}^j$ for κ and $\bar{q}^j := (LDL^T)^{-1}q^j = L^{-T}D^{-1}\hat{q}^j$. Noting that

$$\Lambda(p^{k+1}) = (1-\beta)\left[\Lambda(p^k) + \frac{\beta}{1-\beta}q^j(q^j)^T\right],$$

we see that it is enough to update the Cholesky factorization after a rank-one change, and then scale the diagonal matrix to account for the multiplicative factor. The rank-one update can be performed in an efficient and numerically stable way using the technique of Gill, Murray, and Saunders in [13]. This uses the already computed vector $\hat{q}^j$, and requires $O(n^2)$ operations.

We also need to update the quantities κ_i. Using the update of the matrix $\Lambda(p^k)^{-1}$ in (24), we can easily check that the formulae

$$\kappa_i^+ = \delta(\kappa_i - \sigma((q^i)^T\bar{q}^j)^2/\kappa), \quad i = 1,\ldots,m$$

(where δ and σ are computed as above from $\gamma := 1/\sqrt{\kappa}$) yield the new quantities. (For $i = j$, we use the formula $\kappa_j^+ = \delta(1-\sigma)\kappa$.) Each update of a κ_i requires $O(n)$ operations to perform the inner product, for a total of $O(mn)$. The total complexity of $O(mn)$ operations for each iteration is the same as in Khachiyan [18], but with a more stable implementation. Note that Sun and Freund [28] state that $O(mn^2)$ operations are needed.

In our modified algorithm, we will occasionally subtract a rank-one matrix from $\Lambda(p^k)$ instead of adding one. Here care must be taken to preserve numerical stability; Gill et al. [13] propose a stable technique for this also.

Our implementation recommendation is very similar to that proposed by Goldfarb and Todd [15] for the ellipsoid method. The difference is that here we are proposing maintaining a Cholesky factorization of the possibly sparse matrix $\Lambda(p^k)$, and in most iterations we add a rank-one matrix, while [15] maintained a factorization of $\Lambda(p^k)^{-1}$, and subtracted a rank-one matrix at each iteration.

4 A Modification of the KY Algorithm

Let $\mathcal{A} := \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ be a finite set of vectors whose affine hull is $\mathbb{R}^d$. In this section, we describe a modification of the KY algorithm [21], which, in turn, is derived from Khachiyan's algorithm [18], to compute a feasible solution $\hat{p}$ of $(\mathbf{D}(\mathcal{A}'))$ that satisfies the $(1+\epsilon)$-approximate optimality conditions (20) for any given $\epsilon > 0$.

The main difference between the KY algorithm and Khachiyan's algorithm is the initial feasible solution. The former uses a simple initial volume approximation scheme in an attempt to identify a smaller subset of vectors in $\mathcal{A}$ that provides a reasonable approximation of conv$(\mathcal{A})$. We outline this scheme in the next subsection.

4.1 Initial Volume Approximation

Given $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$, the following deterministic algorithm identifies a subset $\mathcal{X}_0 \subseteq \mathcal{A}$ of size given by $\min\{2d, m\}$ such that vol MVEE$(\mathcal{X}_0)$ is a provable approximation to vol MVEE$(\mathcal{A})$ [21].

Algorithm 4.1 Volume approximation algorithm.

```
Require: Input set of points $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$
 1: If $m \leq 2d$, then $\mathcal{X}_0 \leftarrow \mathcal{A}$. Return.
 2: $\Psi \leftarrow \{0\}$, $\mathcal{X}_0 \leftarrow \emptyset$, $k \leftarrow 0$.
 3: While $\mathbb{R}^d \setminus \Psi \neq \emptyset$, do
 4: loop
 5:     $k \leftarrow k+1$; pick an arbitrary direction $b^k \in \mathbb{R}^d$ in the orthogonal complement of $\Psi$;
 6:     $\alpha \leftarrow \arg\max_{i=1,\ldots,m} (b^k)^T a^i$, $\mathcal{X}_0 \leftarrow \mathcal{X}_0 \cup \{a^\alpha\}$;
 7:     $\beta \leftarrow \arg\min_{i=1,\ldots,m} (b^k)^T a^i$, $\mathcal{X}_0 \leftarrow \mathcal{X}_0 \cup \{a^\beta\}$;
 8:     $\Psi \leftarrow \text{span}(\Psi, \{a^\beta - a^\alpha\})$.
 9: end loop
10: Output $\mathcal{X}_0$.
```

The following lemma provides information about the running time of Algorithm 4.1 and the quality of the resulting approximation.

Lemma 4.1 (Kumar and Yıldırım [21]) *Algorithm 4.1 terminates in $O(md^2)$ time with a subset $\mathcal{X}_0 \subseteq \mathcal{A}$ with $|\mathcal{X}_0| = \min\{2d, m\}$ such that*

$$\textit{vol MVEE}(\mathcal{A}) \leq d^{2d} \textit{vol MVEE}(\mathcal{X}_0). \tag{25}$$

4.2 Our Modification

In this section, we present a modification of the KY algorithm for approximating the minimum volume enclosing ellipsoid of a given set $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$. Given $\epsilon > 0$, we establish that our modification computes an approximate solution that satisfies the ϵ-approximate optimality conditions given by (20). Note that each of the algorithms in [18] and in [21] computes an approximate solution that satisfies the weaker ϵ-relaxed optimality conditions given by (19). In addition, this added benefit does not lead to an increase in the asymptotic complexity result, i.e., the running time of our modified algorithm is asymptotically the same as that of the improved complexity result of the KY algorithm. Finally, we show that our algorithm returns a core set whose asymptotical size has the same bound as that computed by the KY algorithm. In contrast with the KY algorithm which only adds points to the working core set, our modification allows dropping points throughout the algorithm. Therefore, in practice, our modification is likely to compute a smaller core set than that computed by the KY algorithm.

We outline our algorithm below:

Algorithm 4.2 Modified algorithm to compute a feasible solution of $(\mathbf{D}(\mathcal{A}'))$ satisfying (20).

```
Require: Input set of points $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d, \epsilon > 0$.
 1: Run Algorithm 4.1 on $\mathcal{A}$ to get output $\mathcal{X}_0$.
 2: Let $p^0 \in \mathbb{R}^m$ be such that $p^0_i = 1/|\mathcal{X}_0|$ for $a^i \in \mathcal{X}_0$ and $p^0_i = 0$ otherwise.
 3: $k \leftarrow 0$, $n \leftarrow d+1$, and $q^i \leftarrow ((a^i)^T, 1)^T, i = 1, \ldots, m$.
 4: $\mathcal{E}_0 \leftarrow \{y \in \mathbb{R}^n : y^T \Lambda(p^0)^{-1} y \leq 1\}$.
 5: While $p^k$ does not satisfy (20), do
 6: loop
 7:     $j_+ \leftarrow \arg\max\{(q^i)^T \Lambda(p^k)^{-1} q^i : i = 1, \ldots, m\}$, $\kappa_+ \leftarrow (q^{j_+})^T \Lambda(p^k)^{-1} q^{j_+}$;
 8:     $j_- \leftarrow \arg\min\{(q^i)^T \Lambda(p^k)^{-1} q^i : i = 1, \ldots, m,\ p^k_i > 0\}$, $\kappa_- \leftarrow (q^{j_-})^T \Lambda(p^k)^{-1} q^{j_-}$;
 9:     $\epsilon_+ \leftarrow (\kappa_+/n) - 1$, $\epsilon_- \leftarrow 1 - (\kappa_-/n)$;
10:     $\epsilon_k \leftarrow \max\{\epsilon_+, \epsilon_-\}$;
11:     if $\epsilon_k = \epsilon_+$ then
12:         $\beta_k \leftarrow \frac{\kappa_+ - n}{n(\kappa_+ - 1)}$;
13:         $p^{k+1} \leftarrow (1 - \beta_k) p^k + \beta_k e^{j_+}$, $k \leftarrow k + 1$;
14:     else
15:         $\beta_k \leftarrow \min\left\{\frac{n - \kappa_-}{n(\kappa_- - 1)}, \frac{p^k_{j_-}}{1 - p^k_{j_-}}\right\}$;
16:         $p^{k+1} \leftarrow (1 + \beta_k) p^k - \beta_k e^{j_-}$, $k \leftarrow k + 1$;
17:     end if
18:     $\mathcal{E}_k \leftarrow \{y \in \mathbb{R}^n : y^T \Lambda(p^k)^{-1} y \leq 1\}$;
19:     $\mathcal{X}_k \leftarrow \{a^i \in \mathcal{A} : p^k_i > 0\}$.
20: end loop
21: Output $p^k, \mathcal{E}_k$ and $\mathcal{X}_k$.
```

We now describe Algorithm 4.2 in more detail. Given $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$, Algorithm 4.1 is called on $\mathcal{A}$ to get output $\mathcal{X}_0 \subseteq \mathcal{A}$. This subset is used to identify an initial feasible solution p^0 of $(\mathbf{D}(\mathcal{A}'))$. The KY algorithm also uses the same procedure to obtain an initial feasible point. The main difference lies in the execution of each iteration of the main loop. In contrast with the KY algorithm, Algorithm 4.2 computes not only the farthest point $q^{j+} \in \mathcal{A}'$ from the origin in terms of the ellipsoidal norm induced by $\mathcal{E}_k$ but also the closest point $q^{j-} \in \mathcal{A}'$ among those with p_j^k positive. At iteration k,

$$(q^i)^T \Lambda(p^k)^{-1} q^i \leq (1+\epsilon_+)n, \quad i = 1, \ldots, m, \text{ and } (q^i)^T \Lambda(p^k)^{-1} q^i \geq (1-\epsilon_-)n \text{ if } p_i^k > 0 \quad (26)$$

by definition of ϵ_- and ϵ_+, which implies that p^k satisfies the ϵ_k-approximate optimality conditions given by (20). Both Khachiyan's algorithm and the KY algorithm work towards improving ϵ_+. In contrast, Algorithm 4.2 is driven by improving both ϵ_+ and ϵ_-.

Let us now describe how p^k gets updated at iteration k. Since $\mathcal{E}_k \subseteq \mathcal{Q} := \operatorname{conv}\{\pm q^1, \ldots, \pm q^m\}$, p^k is updated in a way to yield the maximum increase in the volume of $\mathcal{E}_{k+1}$. In the case that $\epsilon_k = \epsilon_+$, p^{k+1} is given by a convex combination of p^k and e^{j+}. Since $\log \operatorname{vol}(\mathcal{E}_{k+1})$ is exactly half $\log\det \Lambda(p^{k+1})$, β_k is given by the solution of

$$\beta_k := \arg \max_{\beta \in [0,1]} \log\det \Lambda((1-\beta)p^k + \beta e^{j+}) = \frac{\kappa^+ - n}{n(\kappa^+ - 1)}.$$

Both Khachiyan's algorithm and the KY algorithm use this update. On the other hand, if $\epsilon = \epsilon_-$, then p^{k+1} is obtained from p^k by "moving away" from e^{j-}. (Sun and Freund [28] and Kumar and Yıldırım [21] show that e^{j+} maximizes a linear approximation to the objective function of $(\mathbf{D}(\mathcal{A}'))$ at the current point over the simplex: similarly, it can be seen that e^{j-} minimizes the same linear approximation when restricted only to the positive components of p^k.) In this case, β_k is given by the solution of

$$\beta_k := \arg \max_{\beta \in \left[0, \frac{p^k_{j_-}}{1 - p^k_{j_-}}\right]} \log\det \Lambda((1+\beta)p^k - \beta e^{j-}) = \min\left\{\frac{n - \kappa_-}{n(\kappa_- - 1)}, \frac{p^k_{j_-}}{1 - p^k_{j_-}}\right\}.$$

Note that the range of β is chosen to ensure the feasibility of p^{k+1}.

4.3 Analysis of the Modified Algorithm

Our analysis is based very heavily on those for Khachiyan's and the KY algorithm, but we need to distinguish the three kinds of iterations. If $\epsilon_k = \epsilon_+$, we call the kth iteration a *plus-iteration*. If $\epsilon_k = \epsilon_-$ and $\beta_k = \frac{n-\kappa_-}{n(\kappa_- - 1)}$, we call it a *minus-iteration*. Finally, if $\epsilon_k = \epsilon_-$ and $\beta_k = \frac{p^k_{j_-}}{1-p^k_{j_-}} < \frac{n-\kappa_-}{n(\kappa_- - 1)}$ we call it a *drop-iteration*, because then a component of p^k becomes zero and the associated point a^{j-} is dropped from $\mathcal{X}_k$.

We consider the quantities $\nu_k := \log\det \Lambda(p^k)$ and how they change at each iteration. Note that

$$\Lambda(p^{k+1}) = (1-\alpha)\Lambda(p^k) + \alpha q^j (q^j)^T$$

for $\alpha = \beta_k$ and $j = j_+$ in a plus-iteration, or $\alpha = -\beta_k$ and $j = j_-$ in a minus-iteration or a drop-iteration. It follows that

$$\nu_{k+1} - \nu_k = (n-1)\log(1-\alpha) + \log(1+\alpha(\kappa-1)),$$

where κ is either κ_+ or κ_- respectively. In a drop-iteration, all we can say is that this is nonnegative. (The function above is monotonic until its maximum.) But in a plus- or minus-iteration, we can substitute for the value of α in terms of κ and then κ in terms of δ (either ϵ_+ or $-\epsilon_-$ respectively) to get

$$\begin{array}{rcl} \nu_{k+1} - \nu_k & = & (n-1)\log\left(\frac{(n-1)\kappa}{n(\kappa-1)}\right) + \log\left(\frac{\kappa}{n}\right) \\ & = & (n-1)\log\left(\frac{(n-1)(1+\delta)}{n-1+\delta n}\right) + \log(1+\delta) \\ & = & \log(1+\delta) - (n-1)\log\left(1 + \frac{\delta}{(n-1)(1+\delta)}\right) \\ & \geq & \log(1+\delta) - \frac{\delta}{1+\delta}. \end{array} \tag{27}$$

Lemma 4.2 *In a plus- or a minus-iteration,*

$$\nu_{k+1} - \nu_k \geq \log(1+\epsilon_k) - \frac{\epsilon_k}{1+\epsilon_k}.$$

Proof. This follows directly from (27) in a plus-iteration. To complete the proof, it suffices using (27) again to show that

$$\log(1-\epsilon) + \frac{\epsilon}{1-\epsilon} \geq \log(1+\epsilon) - \frac{\epsilon}{1+\epsilon}$$

for $\epsilon = \epsilon_- = \epsilon_k > 0$. But if we define the functions $f(\epsilon) := \log(1+\epsilon) - \epsilon/(1+\epsilon)$ and $g(\epsilon) := f(-\epsilon)$, we find $f'(\epsilon) = \epsilon/(1+\epsilon)^2$ and $g'(\epsilon) = -f'(-\epsilon) = \epsilon/(1-\epsilon)^2 \geq \epsilon/(1+\epsilon)^2$ for any nonnegative ϵ. Also, f and g are both zero at zero, and this gives $g(\epsilon_k) \geq f(\epsilon_k)$ and hence the result. $\square$

Let us define

$$\tau_0 := \min\{k : \epsilon_k \leq 1\}. \tag{28}$$

Khachiyan's analysis [18] starts by deriving an upper bound on τ_0. To that end, let ν^* denote the optimal value of $(\mathbf{D}(\mathcal{A}'))$. Using Lemma 4.1 and (22), Kumar and Yıldırım [21, Theorem 4.2] establish that $\nu^* - \nu_0 = O(d \log d)$. (Khachiyan's algorithm [18] uses a different initial point which satisfies $\nu^* - \nu_0 = O(d \log m)$, and applies a slightly different argument than that below. This is the reason why the KY algorithm achieves a slightly improved complexity result over Khachiyan's algorithm.) By Lemma 4.2, at each plus- or minus-iteration with $\epsilon_k \geq 1$, we have $\nu_{k+1} - \nu_k \geq \log 2 - 1/2 > 0$. At each drop iteration, we can no longer find a positive lower bound on $\nu_{k+1} - \nu_k \geq 0$. However, each such iteration can be paired with a previous iteration where $p^k_{j_-}$ was increased from zero, except for those where $p^k_{j_-}$ is decreased to zero for the first time and was positive at the initial iteration. Note that the initial feasible point p^0 in Algorithm 4.2 has only $2d$ positive components (in contrast with m positive components in Khachiyan's algorithm). Therefore, doubling the iteration count in the analysis of [21], we find that, after at most $2d + O(d \log d) = O(d \log d)$ iterations, Algorithm 4.2 computes a solution p^k with $\epsilon_k \leq 1$, which implies that

$$\tau_0 = O(d \log d). \tag{29}$$

The next stage of Khachiyan's analysis is aimed at deriving an upper bound on the number of iterations to obtain an iterate p^k with $\epsilon_k \leq \epsilon$. Starting with an iterate with $\epsilon_k \leq 1$, Khachiyan's clever argument [18, Lemma 4] is based on computing an upper bound on the number of iterations to obtain the first iterate with $\epsilon_k \leq 1/2, 1/4, 1/8, \ldots$ It follows from this argument that Khachiyan's algorithm (or the KY algorithm) computes an iterate with $\epsilon_k \leq 1$ after at most $O(d/\epsilon)$ iterations. Due to the possibility of drop-iterations, we can again invoke a similar argument based on iteration doubling and a "fixed charge" (the initial $2d$ positive components of p^k) to establish that Algorithm 4.2 terminates after

$$\tau := \tau_0 + \min\{k : \epsilon_k \leq \epsilon\} = O(d \log d) + O(d/\epsilon) \tag{30}$$

iterations.

Theorem 4.1 *Given $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ and $\epsilon > 0$, Algorithm 4.2 computes a feasible solution p of $(\mathbf{D}(\mathcal{A}'))$ that satisfies the ϵ-approximate optimality conditions given by (20) in $O\left(d[\log d + 1/\epsilon]\right)$ iterations.*

Upon termination of Algorithm 4.2, we have

$$\mathcal{E}_k \subseteq \mathcal{Q} \subseteq \sqrt{(1+\epsilon)n}\,\mathcal{E}_k,$$

where $\mathcal{Q} := \text{conv}\{\pm q^1, \ldots, \pm q^m\}$. Since $\mathcal{Q}$ is centrally symmetric, it follows that $\sqrt{(1+\epsilon)n}\,\mathcal{E}_k \subset \mathbb{R}^n$ is a $(1+\epsilon)n$-rounding of $\mathcal{Q}$. In order to obtain a $(1+\epsilon)d$-rounding of conv$(\mathcal{A})$, it follows from the analysis in [18, Section 3] that Algorithm 4.2 can be called with

$$\epsilon' := \frac{d}{d+1}\epsilon = \frac{d}{n}\epsilon,$$

to obtain $\mathcal{E}_k$. Then let $\mathcal{E} \subset \mathbb{R}^d$ be defined by

$$\mathcal{E} \times \{1\} := \sqrt{(1+\epsilon')n}\,\mathcal{E}_k \cap \Pi = \sqrt{1 + (1+\epsilon)d}\,\mathcal{E}_k \cap \Pi,$$

where Π is given by (11). By [18, Lemma 5],

$$\frac{1}{(1+\epsilon)d}\mathcal{E} \subseteq \text{conv}(\mathcal{A}) \subseteq \mathcal{E},$$

which implies that $\mathcal{E}$ is a $(1+\epsilon)d$-rounding of conv$(\mathcal{A})$.

Corollary 4.1 *Given $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ and $\epsilon > 0$, Algorithm 4.2 computes a $(1+\epsilon)d$-rounding of conv$(\mathcal{A})$ in $O\left(d[\log d + 1/\epsilon]\right)$ iterations.*

So far we have only discussed obtaining $(1+\epsilon)d$-roundings of conv$(\mathcal{A})$, not approximate minimum-volume enclosing ellipsoids. But Khachiyan [18, Theorem 3] shows that a $(1+\eta)$-approximation of MVEE$(\mathcal{A})$ can easily be obtained from a solution to the ϵ-relaxed optimality conditions for $(\mathbf{D}(\mathcal{A}'))$ if $1+\eta = (1+\epsilon)^{n/2}$. Thus to obtain a $(1+\eta)$-approximation of MVEE$(\mathcal{A})$, we merely apply our algorithm with $\epsilon = (1+\eta)^{2/(d+1)} - 1$. From this it is easily seen that our modified algorithm achieves the same complexity (5) as the KY algorithm to obtain a $(1+\eta)$-approximation of MVEE$(\mathcal{A})$, and provides an η-core set $\mathcal{X}$ with the same asymptotic size (6). (Note that each iteration requires $O(md)$ arithmetic operations.)

Corollary 4.2 *Given $\mathcal{A} = \{a^1, \ldots, a^m\} \subset \mathbb{R}^d$ and $\eta > 0$, Algorithm 4.2 computes a $(1+\eta)$-approximation of MVEE$(\mathcal{A})$ in $O\left(md^2\left([(1+\eta)^{2/(d+1)} - 1]^{-1} + \log d\right)\right)$ arithmetic operations and returns an η-core set $\mathcal{X} \subseteq \mathcal{A}$ such that $|\mathcal{X}| = O\left(d[(1+\eta)^{2/(d+1)} - 1]^{-1} + d\log d\right)$.*

5 Final Remarks

In this paper, we have have established a close relationship between Khachiyan's BCD method that computes an approximate rounding of the convex hull of a finite set of vectors and the ellipsoid method using two-sided deepest symmetric cuts. Based on this relationship, we have proposed an efficient way to implement Khachiyan's BCD method. We have also proposed a modification of the KY algorithm that computes an approximate solution to the more complete set of approximate optimality conditions than both the KY and Khachiyan's algorithms. In addition, our algorithm maintains the same asymptotical complexity result and the same bound on core sets as the KY algorithm.

The "dropping idea" in our algorithm that leads to the more complete set of approximate optimality conditions can also be applied in different forms yielding different variants of our algorithm. For instance, if $\epsilon_k = \epsilon_-$ at iteration k of Algorithm 4.2, β_k can be set to its upper bound even if the optimal solution of the linesearch problem defining β_k is in the interior of the range as long as the objective function value does not thereby decrease. This more aggressive dropping technique would lead to the same complexity analysis as that of Algorithm 4.2 and would likely return smaller core sets. We could even consider reducing each positive component of p^k to zero at every iteration, as long as the corresponding $(q^i)^T \Lambda(p^k)^{-1} q^i$ is less than n and the objective function value does not increase, but this would likely be much more expensive.

We finally discuss how to achieve an even smaller core set at the expense of considerable standard linear algebra. The idea is to rewrite $\Lambda(p^k)$ at the end of Algorithm 4.2 as a linear combination of fewer matrices than the number of positive components of p^k. One can find a basic feasible solution p to the system $\sum_{i=1}^m p_i q^i (q^i)^T = \Lambda(p^k), p \geq 0$. (The equation corresponding to the bottom right entry of the matrix ensures that the sum of the components of p is 1.) Such a solution can be computed in $O(md^4)$ operations and would have at most $n(n+1)/2$ positive components, matching the John [17] bound. Indeed, this system has $n(n+1)/2$ rows (note that all matrices are symmetric, so we only need to equate the entries on and above the diagonal) and m columns. We need $O(d^6)$ operations to compute an initial

$O(d^2) \times O(d^2)$ basis inverse, and then at most m iterations requiring $O(d^4)$ operations each to get to a basic feasible solution. Since we are implicitly assuming $m > n(n+1)/2$, the total is $O(md^4)$ operations. However, note that, for constant ϵ and η, this amount of work dominates all the work performed so far, which is $O(md^2(\log d + 1/\epsilon))$ for a $(1+\epsilon)d$-rounding of $\mathcal{A}$, or $O(md^2(\log d + d/\eta))$ for a $(1+\eta)$-approximation of MVEE($\mathcal{A}$).

References

[1] I. Adler and R. Shamir. A randomization scheme for speeding up algorithms for linear and convex quadratic programming problems with a high constraints-to-variables ratio. *Mathematical Programming*, 61:39–52, 1993.

[2] P. K. Aggarwal, R. Poreddy, K. R. Varadarajan, and H. Yu. Practical methods for shape fitting and kinetic data structures using core sets. In *Proceedings of the 20th Annual ACM Symposium on Computaional Geometry*, pages 263–272, 2004.

[3] K. M. Anstreicher. Improved complexity for maximum volume inscribed ellipsoids. *SIAM Journal on Optimization*, 13(2):309–320, 2003.

[4] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38:91–109, 2001.

[5] C. Bouville. Bounding ellipsoid for ray-fractal intersection. In *SIGGRAPH*, pages 45–52, 1985.

[6] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 801–802, 2003.

[7] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

[8] B. P. Burrell and M. J. Todd. The ellipsoid method generates dual variables. *Mathematics of Operations Research*, 10:688–700, 1985.

[9] T. M. Chan. Faster core-set constructions and data stream algorithms in fixed dimensions. In *Proceedings of the 20th Annual ACM Symposium on Computational Geometry*, pages 152–159, 2004.

[10] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. In *Proceedings of the 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 281–290, 1993.

[11] D. Eberly. *3D Game Engine Design.* Morgan Kaufmann, 2001.

[12] B. Gärtner and S. Schönherr. Exact primitives for smallest enclosing ellipses. In *Proceedings of the 13th Annual ACM Symposium on Computational Geometry*, pages 430–432, 1997.

[13] P. E. Gill, W. Murray, and M. A. Saunders. Methods for computing and modifying the LDV factors of a matrix. *Mathematics of Computation*, 29:1051–1077, 1975.

[14] F. Glineur. Pattern separation via ellipsoids and conic programming. Master's thesis, Faculté Polytechnique de Mons, Belgium, 1998.

[15] D. Goldfarb and M. J. Todd. Modifications and implementation of the ellipsoid algorithm for linear programming. *Mathematical Programming*, 23:1–19, 1982.

[16] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, New York, 1988.

[17] F. John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays, presented to R. Courant on his 60th birthday January 8, 1948*, pages 187–204. Interscience, New York, 1948. Reprinted in: *Fritz John, Collected Papers Volume 2* (J. Moser, ed), Birkhäuser, Boston, 1985, pp. 543–560.

[18] L. G. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21:307–320, 1996.

[19] L. G. Khachiyan and M. J. Todd. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61:137–159, 1993.

[20] P. Kumar, J. S. B. Mitchell, and E. A. Yıldırım. Approximate minimum enclosing balls in high dimensions using core-sets. *The ACM Journal of Experimental Algorithmics*, 8(1), 2003.

[21] P. Kumar and E. A. Yıldırım. Minimum volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, 126(1):1–21, 2005.

[22] Jiri Matousek, Micha Sharir, and Emo Welzl. A subexponential bound for linear programming. In *Proceedings of the 8th Annual Symposium on Computational Geometry*, pages 1–8, 1992.

[23] A. S. Nemirovskii. On self–concordant convex–concave functions. *Optimization Methods and Software*, 11/12:303–384, 1999.

[24] Yu. E. Nesterov. Rounding of convex sets and efficient gradient methods for linear programming problems. Discussion Paper 2004-4, CORE, Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2004.

[25] Yu. E. Nesterov and A. S. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming*. SIAM Publications, Philadelphia, 1994.

[26] B. W. Silverman and D. M. Titterington. Minimum covering ellipses. *SIAM Journal on Scientific and Statistical Computing*, 1:401–409, 1980.

[27] S. Silvey and D. Titterington. A geometric approach to optimal design theory. *Biometrika*, 62:21–32, 1973.

[28] P. Sun and R. M. Freund. Computation of minimum volume covering ellipsoids. *Operations Research*, 52:690–706, 2004.

[29] D. M. Titterington. Optimal design: some geometrical aspects of D-optimality. *Biometrika*, 62(2):313–320, 1975.

[30] D. M. Titterington. Estimation of correlation coefficients by ellipsoidal trimming. *Applied Statistics*, 27(3):227–234, 1978.

[31] M. J. Todd. On minimum volume ellipsoids containing part of a given ellipsoid. *Mathematics of Operations Research*, 7:253–261, 1982.

[32] K. C. Toh. Primal-dual path-following algorithms for determinant maximization problems with linear matrix inequalities. *Computational Optimization and Applications*, 14:309–330, 1999.

[33] L. Vandenberghe, S. Boyd, and S. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, 1998.

[34] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In Hermann Maurer, editor, *Proceedings of New Results and New Trends in Computer Science*, volume 555 of *LNCS*, pages 359–370, Berlin, Germany, June 1991. Springer.

[35] Y. Zhang. An interior-point algorithm for the maximum-volume ellipsoid problem. Technical Report TR98-15, Department of Computational and Applied Mathematics, Rice University, 1998.

[36] Y. Zhang and L. Gao. On numerical solution of the maximum volume ellipsoid problem. *SIAM Journal on Optimization*, 14:53–76, 2003.