J Optim Theory Appl (2008) 136: 211–228
DOI 10.1007/s10957-007-9295-9

Computing Minimum-Volume Enclosing Axis-Aligned Ellipsoids

P. Kumar · E.A. Yıldırım

Published online: 14 November 2007

Abstract Given a set of points $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$ and $\epsilon > 0$, we propose and analyze an algorithm for the problem of computing a $(1+\epsilon)$-approximation to the minimum-volume axis-aligned ellipsoid enclosing $\mathcal{S}$. We establish that our algorithm is polynomial for fixed ϵ. In addition, the algorithm returns a small core set $\mathcal{X} \subseteq \mathcal{S}$, whose size is independent of the number of points m, with the property that the minimum-volume axis-aligned ellipsoid enclosing $\mathcal{X}$ is a good approximation of the minimum-volume axis-aligned ellipsoid enclosing $\mathcal{S}$. Our computational results indicate that the algorithm exhibits significantly better performance than the theoretical worst-case complexity estimate.

Keywords Axis-aligned ellipsoids · Enclosing ellipsoids · Core sets · Approximation algorithms

1 Introduction

Real time computer graphics and computer gaming call for the computation of simple bounding regions for rapid culling in the rendering process and for rapid determination that two objects are not intersecting during the collision detection process [1]. An axis-aligned ellipsoid is one such bounding region with low storage complexity that can support fast intersection tests. Currently, in computer graphics, this is done

Communicated by Y. Zhang.

This work was supported in part by the National Science Foundation through CAREER Grants CCF-0643593 and DMI-0237415.

P. Kumar
Department of Computer Science, Florida State University, Tallahassee, FL, USA
e-mail: piyush@cs.fsu.edu

E.A. Yıldırım (✉)
Department of Industrial Engineering, Bilkent University, Bilkent, Ankara, Turkey
e-mail: yildirim@bilkent.edu.tr

by computing the axis-aligned bounding box of the object first and then computing the minimum-volume axis-aligned ellipsoid (MVAE) enclosing the resulting box [1]. While such a scheme usually enables one to quickly compute a reasonably good approximation of the MVAE enclosing the object under consideration, the volume of the resulting ellipsoid may be significantly larger than that of the optimal ellipsoid for certain geometric objects. For instance, if the object is almost spherical, the simple procedure outlined above would return an ellipsoid whose volume may exceed the volume of the MVAE enclosing the object by a factor of as much as $n^{n/2}$, where n is the dimension of the object. Therefore, this procedure may lead to false positive results in collision detection, which provides one of our motivations to study the minimum-volume enclosing axis-aligned ellipsoid problem.

Another application of this problem in higher dimensions is in machine learning, where the kernel approach is widely used [2]. Kernel functions are functions that live in low-dimensional spaces but behave like inner products in high dimensions. In this approach, the main idea is based on *linearization* [3–5], for which kernel functions provide an implicit way. For instance, kernel functions are used in support vector machines to separate nonlinearly separable data via calculating a hyperplane in a different space. In case of enclosing shapes, it is easy to calculate the linearization given a kernel function and then apply optimization algorithms to compute enclosing shapes in the explicitly linearized space. For minimum enclosing balls (MEBs), the kernel version of the problem can be solved very efficiently [6]. The problem with calculating the minimum-volume enclosing ellipsoid (MVEE) in kernel space is that the core set size dependence of MVEEs is quadratic (or higher) in dimension so the number of support vectors produced with such an algorithm is too large for good generalization bounds [2]. Keeping this in mind, a natural question emerges: Is there a bounding shape whose quality is between MEB and MVEE and that can be efficiently computed? Axis-aligned ellipsoids are clearly an answer to this question but efficient algorithms to compute enclosing MVAEs in higher dimensions have not been studied yet. Moreover, for machine learning applications, it is also important to identify a small subset of the input points with the property that the MVAE enclosing this subset is a good approximation of the MVAE enclosing the original set of points. These considerations form a basis for studying minimum-volume enclosing axis-aligned ellipsoids.

In this paper, we propose and study an algorithm that computes an approximation to the MVAE enclosing a given set of m points in $\mathbb{R}^n$. More precisely, given $\mathcal{S} := \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$ and $\epsilon > 0$, our algorithm computes an axis-aligned ellipsoid $\mathcal{E} \subset \mathbb{R}^n$ that satisfies

$$\mathcal{S} \subseteq \mathcal{E}, \quad \text{Vol}\, \mathcal{E} \leq (1+\epsilon)\, \text{Vol}\, \mathcal{E}^*, \tag{1}$$

where $\mathcal{E}^*$ denotes the MVAE enclosing $\mathcal{S}$ and Vol$(\cdot)$ denotes the volume. An axis-aligned ellipsoid $\mathcal{E}$ satisfying (1) is said to be a $(1+\epsilon)$-approximation to the MVAE enclosing $\mathcal{S}$.

Our algorithm is mainly motivated by the algorithm developed earlier by the authors of this paper for the minimum-volume enclosing ellipsoid problem [7] (henceforth the KY algorithm), which, in turn, improves upon the algorithm of Khachiyan [8] by using a simple initial volume approximation scheme. In particular, we establish that our algorithm computes a $(1+\epsilon)$-approximation to the MVAE

enclosing $\mathcal{S}$ in

$$O\left(mn^2\left(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}\right)\right) \tag{2}$$

arithmetic operations (cf. Theorem 4.2), which is linear in m, the number of points in $\mathcal{S}$, and is polynomial for fixed $\epsilon > 0$. In particular, the overall complexity reduces to $O(mn^5/\epsilon)$ for $\epsilon \in (0, 1)$, which implies that our algorithm is especially well-suited for instances of the problem for which $m \gg n$ and for moderately small values of ϵ.

Despite the underlying similarity between the KY algorithm and the algorithm proposed in this paper, our theoretical complexity analysis here is slightly more involved than that of [7] and relies on somewhat different tools. In contrast with the KY algorithm, the one-dimensional line search problem that arises in each iteration of our algorithm does not have a closed form solution. We circumvent this difficulty by using an approximate solution, which, in turn, leads to a worse complexity result given by (2) than that of the KY algorithm. On the other hand, we also establish that our theoretical analysis in general cannot be improved by demonstrating several examples (cf. Sect. 4.2).

Similar to the KY algorithm, our algorithm in this paper also computes a subset $\mathcal{X} \subseteq \mathcal{S}$ with the property that

$$\text{Vol}\,\mathcal{E}^*_{\mathcal{X}} \leq \text{Vol}\,\mathcal{E}^* \leq \text{Vol}\,\mathcal{E} \leq (1+\epsilon)\text{Vol}\,\mathcal{E}^*_{\mathcal{X}} \leq (1+\epsilon)\text{Vol}\,\mathcal{E}^*,$$

where $\mathcal{E}^*_{\mathcal{X}}$ denotes the MVAE enclosing $\mathcal{X}$. It follows from (1) that the ellipsoid $\mathcal{E}$ returned by our algorithm is simultaneously a $(1+\epsilon)$-approximation to the MVAE enclosing $\mathcal{X}$ and to that enclosing $\mathcal{S}$. In addition, $|\mathcal{X}| = O(n(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}))$, which is independent of $|\mathcal{S}| = m$. Following the earlier literature, we call $\mathcal{X}$ an ϵ-core set (or a core set) of $\mathcal{S}$ to signify that $\mathcal{X}$ provides an approximate and compact representation of $\mathcal{S}$. To the best of our knowledge, this establishes the first core set result for the minimum-volume enclosing axis-aligned ellipsoid problem. In comparison with the KY algorithm, the theoretical estimate of the size of the core set for this problem is considerably larger. Similarly to the overall complexity result, this is a byproduct of a more pessimistic theoretical analysis, which, in general, cannot be improved (cf. Sect. 4.2).

In an attempt to highlight the potential discrepancy between the worst-case theoretical complexity result and the practical behavior of the algorithm, we implemented two different versions in MATLAB. While the first version numerically computes an "exact" solution to the line search problem mentioned above, the second one is an exact implementation of our algorithm using only the approximate solution of the line search problem. Our computational experiments reveal that the former version is usually much faster than the latter. These results indicate that the overall computation time and the size of the core set in practice tend to be much smaller than the corresponding worst-case estimates for our algorithm.

The paper is organized as follows. In Sect. 2, we review the optimization formulation of the minimum-volume enclosing axis-aligned ellipsoid problem. Section 3 presents a simple initial volume approximation scheme, which constitutes the initialization stage of our algorithm. In Sect. 4, we present the main algorithm and its analysis. Finally, we conclude the paper in Sect. 5 with future research directions.

2 Optimization Formulation

Let $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$. In this section, we derive optimization formulations for the problem of computing the minimum-volume axis-aligned ellipsoid (MVAE) enclosing $\mathcal{S}$. Throughout the rest of the paper, we will make the following assumption, which ensures that the volume of the optimal enclosing ellipsoid is positive.

Assumption 2.1 The affine hull of $\mathcal{S}$ is $\mathbb{R}^n$.

An axis-aligned ellipsoid $\mathcal{E} \subset \mathbb{R}^n$ is specified by its center $c \in \mathbb{R}^n$ and a positive definite diagonal matrix $D = \operatorname{diag}(d_1, \ldots, d_n) \in \mathbb{R}^{n\times n}$, which determines its shape. Such an ellipsoid can be defined as

$$\mathcal{E} := \{x \in \mathbb{R}^n : (x-c)^T D(x-c) \le 1\} = \left\{ x \in \mathbb{R}^n : \sum_{j=1}^n \left(\sqrt{d_j}[x_j - c_j]\right)^2 \le 1 \right\}.$$

Since the length of each axis is given by $1/\sqrt{d_j}$, $j = 1, \ldots, n$, the volume of $\mathcal{E}$ is given by

$$\operatorname{Vol} \mathcal{E} = \eta_n \det D^{-1/2} = \eta_n \prod_{j=1}^n \frac{1}{\sqrt{d_j}},$$

where η_n is the volume of the unit ball in $\mathbb{R}^n$. We define also a scaled volume by

$$\operatorname{vol} \mathcal{E} = \frac{\operatorname{Vol} \mathcal{E}}{\eta_n} = \det D^{-1/2} = \prod_{j=1}^n \frac{1}{\sqrt{d_j}}.$$

Using a change of variable with $\gamma_j := \sqrt{d_j}$ and $\mu_j := \sqrt{d_j} c_j$, $j = 1, \ldots, n$, and taking the logarithm of the expression for the volume of the enclosing ellipsoid, the problem of computing the MVAE enclosing $\mathcal{S}$ admits the following convex optimization formulation:

$$\begin{aligned} \text{(P)} \quad \min_{\gamma,\mu} \quad & -\sum_{j=1}^n \log \gamma_j, \\ \text{s.t.} \quad & \sum_{j=1}^n \left(\gamma_j x_j^i - \mu_j\right)^2 \le 1, \quad i = 1, \ldots, m, \end{aligned}$$

where $\gamma = (\gamma_1, \ldots, \gamma_n)^T \in \mathbb{R}^n$ and $\mu = (\mu_1, \ldots, \mu_n)^T \in \mathbb{R}^n$.

The Lagrangian dual of (P) is equivalent to the following optimization problem:

$$\begin{aligned} \text{(D)} \quad \max_{\sigma} \quad & \frac{n}{2} \log n + \frac{1}{2} \sum_{j=1}^n \log\left(u_j(\sigma) - v_j^2(\sigma)\right) \\ \text{s.t.} \quad & e^T \sigma = 1, \quad \sigma \ge 0, \end{aligned}$$

where $\sigma \in \mathbb{R}^m$, $e \in \mathbb{R}^m$ denotes the vector of all ones, and

$$u_j(\sigma) := \sum_{i=1}^{m} \sigma_i (x_j^i)^2, \qquad v_j(\sigma) := \sum_{i=1}^{m} \sigma_i x_j^i, \quad j = 1, \ldots, n. \tag{3}$$

It follows from a straightforward but tedious manipulation of the necessary and sufficient optimality conditions for the dual problem (D) that $\sigma^* \in \mathbb{R}^m$ is an optimal solution of (D) if and only if

$$\sum_{j=1}^{n} \frac{(x_j^i - v_j(\sigma^*))^2}{n(u_j(\sigma^*) - v_j^2(\sigma^*))} \le 1, \quad i = 1, \ldots, m, \tag{4a}$$

$$e^T \sigma^* = 1, \tag{4b}$$

$$\sigma_i^* \left(1 - \sum_{j=1}^{n} \frac{(x_j^i - v_j(\sigma^*))^2}{n(u_j(\sigma^*) - v_j^2(\sigma^*))} \right) = 0, \quad i = 1, \ldots, m, \tag{4c}$$

together with $\sigma^* \ge 0$.

Let us define

$$\gamma_j^* := \left[n \left(u_j(\sigma^*) - v_j^2(\sigma^*) \right) \right]^{-1/2}, \qquad \mu_j^* := \gamma_j^* v_j(\sigma^*), \quad j = 1, \ldots, n. \tag{5}$$

Clearly, (γ^*, μ^*) is a feasible solution of (P). Since

$$-\sum_{j=1}^{n} \log \gamma_j^* = \frac{n}{2} \log n + \frac{1}{2} \sum_{j=1}^{n} \log \left(u_j(\sigma^*) - v_j^2(\sigma^*) \right),$$

it follows from strong duality that (γ^*, μ^*) is an optimal solution of (P). As a result, the minimum-volume axis-aligned ellipsoid enclosing $\mathcal{S}$ can be computed by solving the dual problem (D), which will form a basis for our algorithm.

Lemma 2.1 *Let $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$. Then, the minimum-volume enclosing axis-aligned ellipsoid $\mathcal{E}^*$ of $\mathcal{S}$ exists and is unique. Furthermore, if σ^* denotes the optimal solution of* (D), *then $\mathcal{E}^*$ is given by*

$$\mathcal{E}^* := \left\{ x \in \mathbb{R}^n : \sum_{j=1}^{n} \frac{(x_j - v_j(\sigma^*))^2}{n(u_j(\sigma^*) - v_j^2(\sigma^*))} \le 1 \right\}, \tag{6}$$

where $u_j(\cdot)$ and $v_j(\cdot)$ are defined as in (3).

Proof The existence and uniqueness respectively follow from the facts that the feasible region of (D) is compact and that the objective function is strictly concave. The relation (6) is a consequence of the discussions preceding the lemma. □

Undoing the change of variable in (P), it follows from (5) that

$$c_j^* := \frac{\mu_j^*}{\gamma_j^*} = v_j(\sigma^*), \qquad d_j^* := (\gamma_j^*)^2 = \frac{1}{n(u_j(\sigma^*) - v_j^2(\sigma^*))}, \quad j = 1, \ldots, n. \tag{7}$$

Therefore, if σ^* is viewed as a probability measure, c_j^*, $j = 1, \ldots, n$, is simply the expected value of the jth components of the input set and the length of each axis given by $\sqrt{1/d_j^*} = 1/\gamma_j^*$ is proportional to the standard deviation of the jth components of the input set for $j = 1, \ldots, n$ with respect to this probability measure.

Finally, if σ^* is viewed as a set of nonnegative weights for each input point, only the points on the boundary can get a positive weight in an optimal solution by (4c).

3 Initial Volume Approximation

In this section, we present a simple deterministic initial volume approximation algorithm. Given a set of points $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$, the algorithm identifies a small subset $\mathcal{X}_0 \subseteq \mathcal{S}$ such that the minimum volume enclosing axis-aligned ellipsoid of $\mathcal{X}_0$ is closely related to that of $\mathcal{S}$.

Algorithm 3.1 Initial volume approximation algorithm

Step 0. Input: $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$.
Step 1. $k \leftarrow 0$, $\mathcal{X}_0 \leftarrow \emptyset$.
Step 2. While $k < n$, do Steps 3–5 below.
Step 3. $k \leftarrow k + 1$.
Step 4. $\alpha_+ \leftarrow \arg\max_{i=1,\ldots,m} x_k^i$; $\mathcal{X}_0 \leftarrow \mathcal{X}_0 \cup \{x^{\alpha_+}\}$.
Step 5. $\alpha_- \leftarrow \arg\min_{i=1,\ldots,m} x_k^i$; $\mathcal{X}_0 \leftarrow \mathcal{X}_0 \cup \{x^{\alpha_-}\}$.
Step 6. Return $\mathcal{X}_0$.

Lemma 3.1 *Algorithm 3.1 computes a subset $\mathcal{X}_0 \subseteq \mathcal{S}$ with the property that $|\mathcal{X}_0| \leq 2n$ in $O(mn)$ arithmetic operations. Furthermore,*

$$\text{vol } \mathcal{E}^*_{\mathcal{X}_0} \leq \text{vol } \mathcal{E}^* \leq n^{n/2} \text{vol } \mathcal{E}^*_{\mathcal{X}_0}, \tag{8}$$

where $\mathcal{E}^*_{\mathcal{X}_0}$ and $\mathcal{E}^*$ denote the minimum-volume enclosing axis-aligned ellipsoids of $\mathcal{X}_0$ and $\mathcal{S}$, respectively.

Proof Note that Algorithm 3.1 goes through a total of n loops. In each loop, the algorithm computes two points with the minimum and maximum kth coordinates, each of which can be performed in $O(m)$ arithmetic operations, which yields an overall complexity of $O(mn)$ operations.

Let $\mathcal{B}$ denote the minimum-volume axis-aligned enclosing box of $\mathcal{S}$, which coincides with the minimum-volume axis-aligned enclosing box of $\mathcal{X}_0$. Let $\mathcal{E}_{\text{out}}$ denote the

minimum-volume enclosing ellipsoid of $\mathcal{B}$. Since $\mathcal{B}$ is an axis-aligned set, it follows that $\mathcal{E}_{\text{out}}$ is an axis-aligned ellipsoid. Furthermore, since $\mathcal{B}$ is centrally symmetric, it follows from the Löwner-John theorem [9] that

$$\mathcal{E}_{\text{in}} := \frac{1}{\sqrt{n}}\,\mathcal{E}_{\text{out}} \subseteq \mathcal{B} \subseteq \mathcal{E}_{\text{out}}, \tag{9}$$

where $\mathcal{E}_{\text{in}}$ is obtained by scaling $\mathcal{E}_{\text{out}}$ around its center by a factor of $1/\sqrt{n}$. Clearly, vol $\mathcal{E}^*_{\mathcal{X}_0} \le$ vol $\mathcal{E}^* \le$ vol $\mathcal{E}_{\text{out}}$. We now establish that vol $\mathcal{E}_{\text{in}} = (1/\sqrt{n})^n$vol $\mathcal{E}_{\text{out}} \le$ vol $\mathcal{E}^*_{\mathcal{X}_0}$. Since $\mathcal{X}_0 \subseteq \mathcal{E}^*_{\mathcal{X}_0}$, it follows from the projection of each point on each dimension that the length of each axis of $\mathcal{E}^*_{\mathcal{X}_0}$ should be at least as large as the length of the corresponding axis of $\mathcal{E}_{\text{in}}$. Combining these relationships, we obtain

$$\text{vol } \mathcal{E}^*_{\mathcal{X}_0} \le \text{vol } \mathcal{E}^* \le \text{vol } \mathcal{E}_{\text{out}} \le n^{n/2}\text{vol } \mathcal{E}^*_{\mathcal{X}_0},$$

which completes the proof. □

4 Algorithm

In this section, we present and analyze an algorithm (Algorithm 4.1) to compute the minimum-volume enclosing axis-aligned ellipsoid of a given input set of points $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$.

Algorithm 4.1 Minimum-volume enclosing axis-aligned ellipsoid algorithm

Step 0. Input: $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n, \epsilon > 0$.
Step 1. Run Algorithm 3.1 to obtain $\mathcal{X}_0 \subseteq \mathcal{S}$.
Step 2. $\mathcal{X} \leftarrow \mathcal{X}_0$.
Step 3. $\sigma_i^0 \leftarrow 1/|\mathcal{X}_0|$ if $x^i \in \mathcal{X}_0$; $\sigma_i^0 \leftarrow 0$ otherwise.
Step 4. $k \leftarrow 0$; $\quad i_k^* \leftarrow \arg\max_{i=1,\ldots,m} \sum_{j=1}^n \frac{(x_j^i - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))}$.
Step 6. $\epsilon_k \leftarrow \sum_{j=1}^n \frac{(x_j^{i_k^*} - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))} - 1$.
Step 7. While $\epsilon_k > (1+\epsilon)^{2/n} - 1$, do Steps 8–12 below.
Step 8. $\mathcal{X} \leftarrow \mathcal{X} \cup \{x^{i_k^*}\}$.
Step 9. $\beta_k \leftarrow \frac{\epsilon_k}{(n+1)(1+\epsilon_k)}$.
Step 10. $\sigma^{k+1} \leftarrow (1-\beta_k)\sigma^k + \beta_k e^{i_k^*}$; $\quad k \leftarrow k+1$.
Step 11. $i_k^* \leftarrow \arg\max_{i=1,\ldots,m} \sum_{j=1}^n \frac{(x_j^i - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))}$.
Step 12. $\epsilon_k \leftarrow \sum_{j=1}^n \frac{(x_j^{i_k^*} - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))} - 1$.
Step 13. Return $\mathcal{X}$ and $\sqrt{1+\epsilon_k}\,\mathcal{E}_k$, where $\mathcal{E}_k$ is given by (10).

We now describe Algorithm 4.1 in detail. Initially, Algorithm 3.1 is called to compute $\mathcal{X}_0$. The initial iterate σ^0 is obtained by assigning an equal weight to each of

the points in $\mathcal{X}_0$. Note that σ^0 is a feasible solution of (D). At iteration k, a trial axis-aligned ellipsoid $\mathcal{E}_k$ is (implicitly) constructed and is given by

$$\mathcal{E}_k := \left\{ x \in \mathbb{R}^n : \sum_{j=1}^{n} \frac{(x_j - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))} \leq 1 \right\}, \quad k = 0, 1, \ldots, \tag{10}$$

where $u_j(\cdot)$ and $v_j(\cdot)$ are given by (3). Observe that $\mathcal{E}_k$ is the optimal enclosing axis-aligned ellipsoid if and only if σ^k is an optimal solution of (D) (cf. Lemma 2.1). At each iteration, Algorithm 4.1 computes the furthest point $x^{i_k^*}$ from the center of $\mathcal{E}_k$ using its ellipsoidal norm. ϵ_k can be viewed as a quality measure of the kth iterate σ^k (cf. Lemma 4.1). The next iterate σ^{k+1} is given by a convex combination of σ^k and the i_k^*th unit vector $e^{i_k^*}$ with weights $(1 - \beta_k)$ and β_k, respectively, where β_k is computed as in Step 9. This updating scheme simply increases the weight of the furthest point $x^{i_k^*}$ while decreasing those of the remaining points to ensure that the iterate remains feasible for the dual problem (D).

Algorithm 4.1 is a modification of the Frank-Wolfe algorithm [10] applied to the dual optimization problem (D). This idea has previously been used in [7, 8, 11] to compute the minimum-volume enclosing ellipsoid of a given set of points and in [12] to compute the minimum-volume enclosing ellipsoid of a given set of ellipsoids. The Frank-Wolfe algorithm is driven by linearizing the nonlinear objective function of (D) at a given feasible solution σ^k and optimizing this linearized function over the feasible region of (D), which is the unit simplex. Due to the special structure of this feasible region, the optimal solution of the resulting linear programing problem is one of the unit vectors $e^{i_k^*}$. It can easily be shown that i_k^* is precisely the index of the input point that is furthest away from the center of the trial ellipsoid $\mathcal{E}_k$ in its ellipsoidal norm, i.e.,

$$i_k^* := \arg \max_{i=1,\ldots,m} \sum_{j=1}^{n} \frac{(x_j^i - v_j(\sigma^k))^2}{n(u_j(\sigma^k) - v_j^2(\sigma^k))}. \tag{11}$$

The next iterate σ^{k+1} in the Frank-Wolfe algorithm is given by $\sigma^{k+1} := (1 - \beta_k^*)\sigma^k + \beta_k^* e^{i_k^*}$, where β_k^* is given by

$$\beta_k^* := \arg \max_{\beta \in [0,1]} g\left((1 - \beta)\sigma^k + \beta e^{i_k^*}\right), \tag{12}$$

and

$$g(\sigma) := \frac{n}{2} \log n + \frac{1}{2} \sum_{j=1}^{n} \log\left(u_j(\sigma) - v_j^2(\sigma)\right)$$

is the objective function of (D).

In the context of our problem, the main difficulty stems from the fact that the line search problem given in (12) does not have a closed form solution. In general, the optimal solution β_k^* is given by the root in $[0, 1]$ of an nth degree polynomial. The main difference between the Frank-Wolfe algorithm and Algorithm 4.1 is the fact that we do not use the exact minimizer of the line search problem (12). Instead, we

will establish that the particular choice of β_k as computed in Step 9 of Algorithm 4.1 enables us to prove the desired complexity result.

4.1 Analysis of the Algorithm

The complexity analysis of Algorithm 4.1 consists of two main steps. First, we establish an upper bound on the difference of the volume of the initial trial ellipsoid $\mathcal{E}_0$ and that of the optimal ellipsoid $\mathcal{E}^*$. Next, we show that the sequence of the volumes of trial ellipsoids generated by Algorithm 4.1 gives us a sequence of strictly sharper lower bounds on the volume of the optimal ellipsoid $\mathcal{E}^*$.

We start with the following lemma, which establishes the relationship between the volume of each trial ellipsoid $\mathcal{E}_k$ and the volume of the optimal ellipsoid $\mathcal{E}^*$.

Lemma 4.1 *For any $k = 0, 1, \ldots$, we have*

$$\log \operatorname{vol} \mathcal{E}_k \le \log \operatorname{vol} \mathcal{E}^* \le \log \operatorname{vol} \mathcal{E}_k + \frac{n}{2} \log(1 + \epsilon_k), \tag{13}$$

where $\mathcal{E}^*$ denotes the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$.

Proof Note that (cf. (10))

$$\log \operatorname{vol} \mathcal{E}_k = \frac{n}{2} \log n + \frac{1}{2} \sum_{j=1}^{n} \log \left(u_j(\sigma^k) - v_j(\sigma^k) \right).$$

The first inequality follows from the fact that σ^k is a feasible solution of the dual problem for all $k = 0, 1, \ldots$, and that $\log \operatorname{vol} \mathcal{E}^*$ coincides with the optimal value of the dual problem.

By definition of ϵ_k, we have $\mathcal{S} \subseteq \sqrt{1 + \epsilon_k}\, \mathcal{E}_k$, which implies that $\log \operatorname{vol} \mathcal{E}^* \le \log \operatorname{vol} \mathcal{E}_k + (n/2) \log(1 + \epsilon_k)$, proving the second inequality. □

An immediate corollary of Lemma 4.1 is that $\epsilon_k \ge 0$ for all $k = 0, 1, \ldots$ and $\epsilon_k = 0$ if and only if σ^k is an optimal solution of (D).

The next lemma provides a bound on the volume of the initial trial ellipsoid $\mathcal{E}_0$.

Lemma 4.2 *Let $\mathcal{E}_0$ denote the trial ellipsoid corresponding to σ^0. Then,*

$$\log \operatorname{vol} \mathcal{E}_0 \le \log \operatorname{vol} \mathcal{E}^* \le \log \operatorname{vol} \mathcal{E}_0 + n \log n + (n/2) \log 2. \tag{14}$$

Proof The first inequality in (14) is a direct consequence of Lemma 4.1. In order to prove the second inequality, let us define $\mathcal{I} := \{i \in \{1, \ldots, m\} : x^i \in \mathcal{X}_0\}$. For $i \in \mathcal{I}$, let

$$z_i := \sum_{j=1}^{n} \frac{(x_j^i - v_j(\sigma^0))^2}{n(u_j(\sigma^0) - v_j^2(\sigma^0))}, \tag{15}$$

i.e., z_i is the squared distance of x^i from the center of $\mathcal{E}_0$ measured in terms of its ellipsoidal norm for $i \in \mathcal{I}$. Clearly, $z_i \geq 0,\ i \in \mathcal{I}$. By (3) and the definition of σ^0, we have

$$\begin{aligned}\sum_{i\in\mathcal{I}} z_i &= \sum_{i\in\mathcal{I}}\sum_{j=1}^{n} \frac{(x_j^i - v_j(\sigma^0))^2}{n(u_j(\sigma^0) - v_j^2(\sigma^0))},\\ &= \sum_{j=1}^{n} \frac{|\mathcal{X}_0|(u_j(\sigma^0) - v_j^2(\sigma^0))}{n(u_j(\sigma^0) - v_j^2(\sigma^0))},\\ &= |\mathcal{X}_0|.\end{aligned}$$

Note that

$$\max_{i\in\mathcal{I}} z_i \leq \sum_{i\in\mathcal{I}} z_i = |\mathcal{X}_0| \leq 2n,$$

which implies that $\mathcal{X}_0 \subseteq \sqrt{2n}\,\mathcal{E}_0$. Therefore, $\log\text{vol}\,\mathcal{E}^*_{\mathcal{X}_0} \leq \log\text{vol}\,\mathcal{E}_0 + (n/2)\log n + (n/2)\log 2$, where $\mathcal{E}^*_{\mathcal{X}_0}$ is the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{X}_0$. By Lemma 3.1, $\log\text{vol}\,\mathcal{E}^* \leq \log\text{vol}\,\mathcal{E}^*_{\mathcal{X}_0} + (n/2)\log n$. Combining these two inequalities, we obtain $\log\text{vol}\,\mathcal{E}^* \leq \log\text{vol}\,\mathcal{E}_0 + n\log n + (n/2)\log 2$, which establishes the second inequality in (14). □

We now reconsider the line search problem given by (12). We are interested in

$$\max_{\beta\in[0,1]} \Psi_k(\beta), \quad k = 0, 1, \ldots, \tag{16}$$

where

$$\begin{aligned}\Psi_k(\beta) &:= g\left((1-\beta)\sigma^k + \beta e^{i_k^*}\right)\\ &= \frac{n}{2}\log n + \frac{1}{2}\sum_{j=1}^{n}\log\left(u_j\left((1-\beta)\sigma^k + \beta e^{i_k^*}\right) - v_j^2\left((1-\beta)\sigma^k + \beta e^{i_k^*}\right)\right)\\ &= \frac{n}{2}\log n\\ &\quad + \frac{1}{2}\sum_{j=1}^{n}\log\left((1-\beta)u_j(\sigma^k) + \beta\left(x_j^{i_k^*}\right)^2 - \left[(1-\beta)v_j(\sigma^k) + \beta x_j^{i_k^*}\right]^2\right)\\ &= \frac{n}{2}\log n + \frac{n}{2}\log(1-\beta)\\ &\quad + \frac{1}{2}\sum_{j=1}^{n}\log\left(u_j(\sigma^k) - v_j^2(\sigma^k) + \beta\left[x_j^{i_k^*} - v_j(\sigma^k)\right]^2\right)\\ &= \frac{n}{2}\log n + \frac{1}{2}\sum_{j=1}^{n}\log\left(u_j(\sigma^k) - v_j^2(\sigma^k)\right)\end{aligned}$$

$$+\frac{n}{2}\log(1-\beta)+\frac{1}{2}\sum_{j=1}^{n}\log\left(1+\frac{\beta[x_j^{i_k^*}-v_j(\sigma^k)]^2}{u_j(\sigma^k)-v_j^2(\sigma^k)}\right)$$

$$=g(\sigma^k)+\Delta_k(\beta),$$

where

$$\Delta_k(\beta):=\frac{n}{2}\log(1-\beta)+\frac{1}{2}\sum_{j=1}^{n}\log\left(1+\frac{\beta[x_j^{i_k^*}-v_j(\sigma^k)]^2}{u_j(\sigma^k)-v_j^2(\sigma^k)}\right),\quad k=0,1,\ldots. \tag{17}$$

Therefore, the line search problem (16) is equivalent to

$$\max_{\beta\in[0,1]}\Delta_k(\beta),\quad k=0,1,\ldots. \tag{18}$$

Let us define

$$w_j(\sigma^k):=\frac{(v_j(\sigma^k)-x_j^{i_k^*})^2}{n(u_j(\sigma^k)-v_j^2(\sigma^k))},\quad j=1,\ldots,n,\ k=0,1,\ldots. \tag{19}$$

By definition of ϵ_k (cf. Step 12 of Algorithm 4.1), we have

$$1+\epsilon_k=\sum_{j=1}^{n}w_j(\sigma^k),\quad k=0,1,\ldots. \tag{20}$$

It follows from (17) and (19) that the line search problem (18) can be rewritten as

$$\max_{\beta\in[0,1]}\Delta_k(\beta)=\max_{\beta\in[0,1]}\left\{\frac{n}{2}\log(1-\beta)+\frac{1}{2}\sum_{j=1}^{n}\log\left(1+\beta n w_j(\sigma^k)\right)\right\},$$

$$k=0,1,\ldots. \tag{21}$$

The next lemma shows that this optimization problem has a unique solution and provides useful information on the unique maximizer.

Lemma 4.3 *For each $k=0,1,\ldots$, the function $\Delta_k(\beta)$ has a unique maximizer $\beta_k^*\in[0,1)$. Furthermore,*

$$\beta_k^*\geq\beta_k:=\frac{\epsilon_k}{(n+1)(1+\epsilon_k)},\quad k=0,1,\ldots, \tag{22}$$

where ϵ_k is defined as in Step 12 of Algorithm 4.1.

Proof We first prove that $\Delta_k(\beta)$ is a strictly concave function on $[0,1)$. Taking the derivative, we obtain

$$\Delta_k'(\beta)=-\frac{n}{2(1-\beta)}+\frac{1}{2}\sum_{j=1}^{n}\frac{nw_j(\sigma^k)}{1+\beta n w_j(\sigma^k)}. \tag{23}$$

By (23),

$$\Delta_k''(\beta) = -\frac{n}{2(1-\beta)^2} - \frac{1}{2}\sum_{j=1}^{n} \frac{n^2 w_j^2(\sigma^k)}{1+\beta n w_j(\sigma^k)},$$

which is clearly negative on $[0, 1)$. Since $\Delta_k'(0) = n\epsilon_k/2 \geq 0$ by (20), $\lim_{\beta\uparrow 1} \Delta'(\beta) = -\infty$, and $\Delta'(\beta)$ is strictly decreasing, it follows that there exists a unique $\beta_k^* \in [0, 1)$ such that $\Delta'(\beta_k^*) = 0$, which completes the first part of the proof.

By (23), we have

$$\begin{aligned}\Delta_k'(\beta) &\geq \frac{1}{2}\left(-\frac{n}{1-\beta} + \sum_{j=1}^{n} \frac{n w_j(\sigma^k)}{1+\beta n(1+\epsilon_k)}\right) \\ &= \frac{1}{2}\left(-\frac{n}{1-\beta} + \frac{n(1+\epsilon_k)}{1+\beta n(1+\epsilon_k)}\right),\end{aligned}$$

where we used $w_j(\sigma^k) \leq 1+\epsilon_k$ to derive the first inequality and (20) in the last equality. It follows from this inequality that

$$\Delta_k'(\beta_k) \geq \frac{1}{2}\left(-\frac{n}{1-\beta_k} + \frac{n(1+\epsilon_k)}{1+\beta_k n(1+\epsilon_k)}\right) = 0 = \Delta_k'(\beta_k^*).$$

Since $\Delta_k'(\beta)$ is a strictly decreasing function, it follows that $\beta_k^* \geq \beta_k$, which concludes the second part of the proof. □

Lemma 4.3 establishes that β_k used in Step 9 of Algorithm 4.1 is a lower bound on the unique solution β_k^* of the line search problem (21). It follows from (23) that solving the equation $\Delta_k'(\beta) = 0$ is equivalent to finding a zero of an nth degree polynomial in $[0, 1]$. As such, β_k^* does not have a closed form solution. This is the reason why Algorithm 4.1 differs from the Frank-Wolfe algorithm by employing β_k as opposed to β_k^*. The next lemma establishes a lower bound on the improvement of the objective function of the dual problem (D) in each iteration using this particular choice of β_k.

Lemma 4.4 *Let β_k^* be the unique maximizer of $\Delta_k(\beta)$ in $[0, 1)$ and let δ_k be given by*

$$\delta_k := \max_{j=1,\ldots,n} \beta_k n w_j(\sigma^k), \quad k = 0, 1, \ldots, \tag{24}$$

where β_k and $w_j(\cdot)$ are defined by (22) and (19), respectively. Then, for $k = 0, 1, \ldots$,

$$\Delta_k(\beta_k^*) \geq \Delta_k(\beta_k) \geq \begin{cases} \frac{1}{2}\log 2 - \frac{1}{4} > 0, & \text{if } \delta_k \geq 1, \\ \frac{\delta_k^2}{16}, & \text{otherwise.} \end{cases} \tag{25}$$

Proof Since $\Delta_k'(\beta_k) \geq 0$ by the proof of Lemma 4.3, it follows from (23) that

$$\sum_{j=1}^{n} \frac{w_j(\sigma^k)}{1+\beta_k n w_j(\sigma^k)} \geq \frac{1}{1-\beta_k}. \tag{26}$$

Since β_k^* is the maximizer of $\Delta_k(\beta)$ in $[0, 1)$, we have

$$\begin{aligned}\Delta_k(\beta_k^*) &\geq \Delta_k(\beta_k)\\ &= \frac{n}{2}\log(1-\beta_k) + \frac{1}{2}\sum_{j=1}^{n}\log\left(1+\beta_k n w_j(\sigma^k)\right)\\ &= -\frac{n}{2}\log\left(1+\frac{\beta_k}{1-\beta_k}\right) + \frac{1}{2}\sum_{j=1}^{n}\log\left(1+\beta_k n w_j(\sigma^k)\right)\\ &\geq -\frac{n\beta_k}{2(1-\beta_k)} + \frac{1}{2}\sum_{j=1}^{n}\log\left(1+\beta_k n w_j(\sigma^k)\right)\\ &\geq -\frac{1}{2}\sum_{j=1}^{n}\frac{\beta_k n w_j(\sigma^k)}{1+\beta_k n w_j(\sigma^k)} + \frac{1}{2}\sum_{j=1}^{n}\log\left(1+\beta_k n w_j(\sigma^k)\right),\end{aligned}$$

where we used $\log(1+x) \leq x$ for $x > -1$ and (26) to derive the second and the third inequalities, respectively.

It is easy to verify that the function $\Xi(x) := \log(1+x) - x/(1+x)$ is a strictly increasing function for $x \geq 0$ and $\Xi(x) \geq (1/8)x^2$ for $x \in [0, 1)$. Therefore, if $\delta_k \geq 1$, it follows from the above inequality that

$$\Delta_k(\beta_k) \geq \frac{1}{2}\log 2 - \frac{1}{4}.$$

Otherwise, we obtain

$$\Delta_k(\beta_k) \geq \frac{\delta_k^2}{16},$$

which concludes the proof. □

Note that the lower bound on the improvement is presented in terms of the value of δ_k in Lemma 4.4. The next lemma relates this quantity to ϵ_k, which is the parameter used to determine the convergence of Algorithm 4.1.

Lemma 4.5 *Let δ_k be given by (24). Then,*

$$\epsilon_k \leq \delta_k(n+1), \quad k = 0, 1, \ldots. \tag{27}$$

Proof Clearly, $\beta_k n w_j(\sigma^k) \leq \delta_k$, $j = 1, \ldots, n$. Summing these inequalities over $j = 1, \ldots, n$, we obtain $\beta_k n(1+\epsilon_k) \leq \delta_k n$, where we used (20). Solving for ϵ_k, we obtain the desired inequality (27) using (22). □

Let us now define

$$\kappa_r := \min\left\{k : \delta_k \leq \frac{1}{2^r}\right\}, \quad r = 0, 1, \ldots. \tag{28}$$

The next lemma plays a key role in the complexity analysis.

Lemma 4.6 *Let $n \geq 2$. The following relationships are satisfied*:

$$\kappa_0 = O(n \log n), \tag{29}$$

$$\kappa_{r+1} - \kappa_r \leq 2^{r+6} n^2, \quad r = 0, 1, \ldots. \tag{30}$$

Proof By Lemma 4.2, $\log \text{vol}\, \mathcal{E}_0 \leq \log \text{vol}\, \mathcal{E}^* \leq \log \text{vol}\, \mathcal{E}_0 + n \log n + (n/2) \log 2 \leq \log \text{vol}\, \mathcal{E}_0 + 2n \log n$ for $n \geq 2$. At each iteration k with $\delta_k > 1$, $\log \text{vol}\, \mathcal{E}_{k+1} - \log \text{vol}\, \mathcal{E}_k = \Delta_k(\beta_k) \geq (1/2) \log 2 - 1/4 > 0$ by Lemma 4.4. Therefore, $\kappa_0 = O(n \log n)$, which proves (29).

In order to prove (30), let $\rho := \kappa_r$. Then, $\epsilon_\rho \leq \delta_\rho(n+1) \leq (1/2^r)(n+1)$ by Lemma 4.5 and by the definition of ρ. By Lemma 4.1, $\log \text{vol}\, \mathcal{E}_\rho \leq \log \text{vol}\, \mathcal{E}^* \leq \log \text{vol}\, \mathcal{E}_\rho + (n/2) \log(1 + \epsilon_\rho) \leq \log \text{vol}\, \mathcal{E}_\rho + (n/2)\epsilon_\rho \leq \log \text{vol}\, \mathcal{E}_\rho + (1/2^{r+1})n(n+1)$, where we used $\log(1+x) \leq x$ for $x > -1$. At each iteration k with $\delta_k > 1/2^{r+1}$, we have $\log \text{vol}\, \mathcal{E}_{k+1} - \log \text{vol}\, \mathcal{E}_k = \Delta_k(\beta_k) \geq 1/2^{2r+2+4} = 1/2^{2r+6}$ by Lemma 4.4. Therefore, $\kappa_{r+1} - \kappa_r \leq ((1/2^{r+1})n(n+1))/(1/2^{2r+6}) = 2^{r+5} n(n+1) \leq 2^{r+6} n^2$, where we used $n + 1 \leq 2n$. This completes the proof. □

The next lemma gives an upper bound on the number of iterations to obtain an iterate σ^k with $\delta_k \leq \nu$.

Lemma 4.7 *Let $\nu \in (0, 1)$. Then, Algorithm 4.1 computes an iterate with $\delta_k \leq \nu$ in $O(n \log n + n^2/\nu)$ iterations.*

Proof Let p be an integer such that $1/2^{p+1} \leq \nu \leq 1/2^p$. Then, after $k = \kappa_{p+1}$ iterations, we already have $\delta_k \leq 1/2^{p+1} \leq \nu$. By Lemma 4.6, we have

$$\kappa_{p+1} = \kappa_0 + \sum_{r=0}^{p} (\kappa_{r+1} - \kappa_r) \leq \kappa_0 + \sum_{r=0}^{p} 2^{r+6} n^2 \leq \kappa_0 + 64 n^2 2^{p+1}$$

$$= O\left(n \log n + \frac{n^2}{\nu}\right),$$

where we used $2^{p+1} \leq 2/\nu$. □

We now have all the ingredients to establish the iteration complexity of Algorithm 4.1.

Theorem 4.1 *Algorithm 4.1 computes a $(1+\epsilon)$-approximation to the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$ in $O(n(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}))$ iterations.*

Proof We first establish that it suffices to run Algorithm 4.1 until we obtain an iterate with $\delta_k \leq [(1+\epsilon)^{2/n} - 1]/(n+1)$. Let k^* denote the index of first such iterate. Then, $\epsilon_{k^*} \leq \delta_{k^*}(n+1) \leq (1+\epsilon)^{2/n} - 1$, which implies that the termination criterion is

satisfied. It follows then that $1+\epsilon_{k^*} \leq (1+\epsilon)^{2/n}$. However, $\mathcal{S} \subseteq \sqrt{1+\epsilon_{k^*}}\,\mathcal{E}_{k^*}$. By Lemma 4.1,

$$\begin{aligned}\operatorname{vol} \mathcal{E}_{k^*} &\leq \operatorname{vol} \mathcal{E}^* \leq \operatorname{vol} \sqrt{1+\epsilon_{k^*}}\,\mathcal{E}_{k^*} \\ &= (1+\epsilon_{k^*})^{n/2} \operatorname{vol} \mathcal{E}_{k^*} \leq (1+\epsilon) \operatorname{vol} \mathcal{E}_{k^*} \leq (1+\epsilon) \operatorname{vol} \mathcal{E}^*. \end{aligned} \tag{31}$$

Therefore, $\sqrt{1+\epsilon_{k^*}}\,\mathcal{E}_{k^*}$ is a $(1+\epsilon)$-approximation to the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$.

By Lemma 4.7, Algorithm 4.1 computes such an iterate in

$$\begin{aligned} &O\left(n \log n + \left((n+1)n^2\right) / \left[(1+\epsilon)^{2/n} - 1\right]\right) \\ &= O\left(n\left(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}\right)\right) \end{aligned}$$

iterations. □

The next theorem establishes the overall complexity of Algorithm 4.1.

Theorem 4.2 *Algorithm 4.1 computes a $(1+\epsilon)$-approximation to the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$ in $O(mn^2(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}))$ arithmetic operations.*

Proof Algorithm 4.1 first calls Algorithm 3.1, which computes the initial iterate σ^0 in $O(mn)$ operations by Lemma 3.1. Note that each of the linear functions $u_j(\cdot)$ and $v_j(\cdot)$ can be updated in constant time since $\sigma^{k+1} = (1-\beta_k)\sigma^k + \beta_k e^{i_k^*}$. Therefore, i_k^* can be computed in $O(mn)$ time. Combining this result with Theorem 4.1 establishes the assertion. □

Remark 4.1 Theorem 4.2 establishes the overall complexity of Algorithm 4.1. We stress that the complexity result depends linearly on m, the number of points. In addition, for $\epsilon \in (0,1)$, the complexity of Algorithm 4.1 is given by $O(mn^5/\epsilon)$ since $[(1+\epsilon)^{2/n} - 1]^{-1} = O(n/\epsilon)$. Therefore, from a theoretical point of view, this suggests that Algorithm 4.1 is especially well-suited for instances of the minimum-volume enclosing axis-aligned ellipsoid problem with $m \gg n$ and for moderately small values of ϵ. Furthermore, the complexity result is polynomial for fixed $\epsilon > 0$.

Finally, we establish the following core set result.

Theorem 4.3 *Let k^* denote the index of the final iterate computed by Algorithm 4.1. Let $\mathcal{E}^*_{\mathcal{X}_{k^*}}$ and $\mathcal{E}^*$ denote the minimum-volume enclosing axis-aligned ellipsoids of $\mathcal{X}_{k^*}$ and $\mathcal{S}$, respectively. Then,*

$$\operatorname{vol} \mathcal{E}^*_{\mathcal{X}_{k^*}} \leq \operatorname{vol} \mathcal{E}^* \leq (1+\epsilon) \operatorname{vol} \mathcal{E}^*_{\mathcal{X}_{k^*}}. \tag{32}$$

Furthermore,

$$|\mathcal{X}_{k^*}| = O\left(n\left(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}\right)\right). \tag{33}$$

Proof The first inequality in (32) is obvious since $\mathcal{X}_{k^*} \subseteq \mathcal{S}$. The second inequality is a consequence of the inequalities vol $\mathcal{E}_{k^*} \leq$ vol $\mathcal{E}^*_{\mathcal{X}_{k^*}} \leq$ vol $\mathcal{E}^*$ (cf. the proof of Lemma 4.1) and vol $\mathcal{E}^* \leq (1+\epsilon)$vol $\mathcal{E}_{k^*}$ (cf. the proof of Theorem 4.1).

Clearly, $|\mathcal{X}_{k^*}| \leq 2n + k^* = O(n(\log n + n^2[(1+\epsilon)^{2/n} - 1]^{-1}))$ by Theorem 4.1. □

Remark 4.2 Theorem 4.3 establishes a core set result with the property that the size of the core set is independent of m, the number of points. To the best of our knowledge, this is the first core set result established for the minimum-volume enclosing axis-aligned ellipsoid problem. As such, it is an addition to the previous core set results established for similar geometric optimization problems such as the minimum enclosing ball problem [13], the minimum volume enclosing ellipsoid problem [7, 11] and the minimum volume enclosing ellipsoid of ellipsoids problem [12]. For $\epsilon \in (0, 1)$, the size of the core set is given by $O(n^4/\epsilon)$.

4.2 Justification of the Larger Core Set Size

Given a set of points $\mathcal{S} = \{x^1, \ldots, x^m\} \subset \mathbb{R}^n$, note that the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$ is determined by a subset $\mathcal{X} \subseteq \mathcal{S}$ with at most $2n$ points since one needs to compute a total of $2n$ parameters corresponding to the center and the length of each axis of the minimum-volume enclosing axis-aligned ellipsoid of $\mathcal{S}$. Therefore, in theory, there always exists a core set whose size is $O(n)$.

Our analysis of Algorithm 4.1 establishes a core set size of $O(n^4/\epsilon)$ for $\epsilon \in (0, 1)$ (cf. Theorem 4.3). In this subsection, we attempt to justify this discrepancy. In particular, we will argue that the theoretical analysis is rather pessimistic. On the other hand, we provide simple examples illustrating that the analysis in general cannot be improved.

A close examination of the analysis of Algorithm 4.1 reveals two potential sources which lead to a larger core set size in comparison with the core set size of $O(n^2/\epsilon)$ established for the minimum-volume enclosing ellipsoid problem in [7, 11].

The first potential source of the larger core set size arises from the line search problem (12). As we discussed in Sect. 4, Algorithm 4.1 differs from the algorithm proposed in [7] in the sense that it employs the lower bound β_k as opposed to the exact maximizer β_k^* of the line search problem (cf. Lemma 4.3) since β_k^* does not have a closed form solution. From a computational point of view, β_k^* can be computed to within an arbitrary precision using, for instance, binary search since Lemma 4.3 establishes that the function $\Delta_k'(\beta)$ is strictly decreasing and has a unique zero in $(0, 1)$. Note that this computation only adds a fixed overhead at each iteration of Algorithm 4.1. However, the theoretical analysis heavily depends on establishing a lower bound on the improvement given by $\Delta_k(\beta)$ at each step of Algorithm 4.1. Our improvement result relies on the lower bound β_k. Clearly, using the lower bound β_k instead of the exact maximizer β_k^* yields a smaller improvement in general.

On the other hand, the following example establishes that, for certain data sets, β_k can coincide with β_k^* for some iterations of Algorithm 4.1. Therefore, the lower bound β_k on β_k^* used in the analysis in general cannot be improved.

Example 4.1 It is easy to verify that if $w_j(\sigma^k) = 1 + \epsilon_k$ and $w_l(\sigma^k) = 0$ for $l = 1, \ldots, n; l \neq j$, then $\beta_k = \beta_k^*$. This simple example illustrates that this can

indeed happen. Let $\mathcal{S} := \{x^1, \ldots, x^4\} \subset \mathbb{R}^2$, where $x^1 = (1, 0)^T$, $x^2 = (-1, 0)^T$, $x^3 = (0, -1)^T$ and $x^4 = (0, 3)^T$. Then, running Algorithm 3.1 yields $\mathcal{X}_0 = \mathcal{S} = \{x^1, \ldots, x^4\}$ and $\sigma^0 = (1/4, 1/4, 1/4, 1/4)^T$. By (3), $u_1(\sigma^0) = 1/2$, $u_2(\sigma^0) = 5/2$, $v_1(\sigma^0) = 0$, and $v_2(\sigma^0) = 1/2$. Therefore, the initial trial ellipsoid $\mathcal{E}_0$ is given by $\mathcal{E}_0 := \{x \in \mathbb{R}^2 : x_1^2 + (2/9)(x_2 - 1/2)^2 \leq 1\}$ (cf. (10)). It is easy to verify that the furthest point in $\mathcal{S}$ from the center of $\mathcal{E}_0$ in its ellipsoidal norm is x^4, which implies that $i_0^* = 4$. By (19), we have $w_1(\sigma^0) = 25/18$ and $w_2(\sigma^0) = 0$. Therefore, $\beta_0^* = \beta_0 = [(25/18) - 1]/[3(25/18)] = 7/75$.

The second potential source of the larger core set size is the relationship between δ_k and ϵ_k given by Lemma 4.5. For instance, if $w_1(\sigma^k) = 1 + \epsilon_k$ and $w_j(\sigma^k) = 0$ for $j = 2, \ldots, n$ (cf. Example 4.1), then it is easy to verify that $\delta_k = \beta_k n w_1(\sigma^k) = (\epsilon_k/[(n + 1)(1 + \epsilon_k)])n(1 + \epsilon_k) = [n/(n + 1)]\epsilon_k$. Using the relationship given by Lemma 4.5, we obtain $\epsilon_k \leq \delta_k(n + 1) = n\epsilon_k$, which implies that the upper bound on ϵ_k can be worse by a factor of n. However, the next example illustrates that this upper bound in general cannot be improved, either.

Example 4.2 If $w_j(\sigma^k) = (1 + \epsilon_k)/n$ for $j = 1, \ldots, n$, one can verify that $\epsilon_k = \delta_k(n + 1)$, i.e., the inequality given by Lemma 4.5 is satisfied with equality. We provide a simple example illustrating that this can indeed happen. Let $\mathcal{S} = \{x^1, \ldots, x^5\} \subset \mathbb{R}^2$, where $x^1 = (1, 0)^T$, $x^2 = (0, 1)^T$, $x^3 = (-1, 0)^T$, $x^4 = (0, -1)^T$, and $x^5 = (.9, .9)^T$. Then, running Algorithm 3.1 yields $\mathcal{X}_0 = \{x^1, \ldots, x^4\}$ and $\sigma^0 = (1/4, 1/4, 1/4, 1/4, 0)^T$. By (3), $u_1(\sigma^0) = u_2(\sigma^0) = 1/2$, and $v_1(\sigma^0) = v_2(\sigma^0) = 0$. Therefore, the initial trial ellipsoid $\mathcal{E}_0$ is given by $\mathcal{E}_0 := \{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \leq 1\}$ (cf. (10)). Clearly, x^5 is the only point which lies outside $\mathcal{E}_0$, which implies that $i_0^* = 5$. By (19), we have $w_1(\sigma^0) = w_2(\sigma^0) = .81$. By (20), $\epsilon_0 = 2(.81) - 1 = .62$. Similarly, $\delta_0 = \max_{j=1,2} \beta_0 n w_j(\sigma^0) = (.62/(3(1.62)))2(.81) = .62/3$, which implies that $\epsilon_0 = 3\delta_0$.

These small examples illustrate that our theoretical analysis of Algorithm 4.1 in general cannot be improved. In addition, they help to explain the larger core set size presented in Theorem 4.3.

On the other hand, it is reasonable to expect that situations arising in Example 4.1 and 4.2 will occur fairly infrequently for general data sets. In fact, our preliminary computational results support our claim in the sense that both the core set size and the running time in practice tend to be far smaller than those predicted by the theoretical analysis.

5 Concluding Remarks

In this paper, we proposed and analyzed an algorithm to approximately compute the minimum-volume enclosing axis-aligned ellipsoid of a given set of points. We established the existence of a core set whose size is independent of the number of points. As illustrated by several examples and our computational results, the theoretical analysis is rather pessimistic but cannot be improved in general.

There are many interesting theoretical and practical problems that are motivated by our investigations. In the near future, we intend to study the minimum-volume enclosing axis-aligned ellipsoid problem with outliers and the k-center problem using axis-aligned ellipsoids.

References

1. Eberly, D.: 3D Game Engine Design. Kaufmann, San Francisco (2001)
2. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
3. Yao, A.C., Yao, F.F.: A general approach to d-dimensional geometric queries. In: Proceedings of 17th Annual ACM Symposium on Theory of Computing, pp. 163–168 (1985)
4. Agarwal, P.K., Matoušek, J.: On range searching with semialgebraic sets. In: Proceedings of 17th International Symposium on Mathematical Foundations of Computer Science. Lecture Notes in Computer Science, vol. 629, pp. 1–13. Springer, New York (1992)
5. Matoušek, J., Schwarzkopf, O.: A deterministic algorithm for the three-dimensional diameter problem. In: Proceedings of 25th Annual ACM Symposium on Theory of Computing, pp. 478–484 (1993)
6. Bulatov, Y., Jambawalikar, S., Kumar, P., Sethia, S.: Hand recognition using geometric classifiers. In: Proceedings of International Conference on Biometric Authentication. Lecture Notes in Computer Science, vol. 3072, pp. 753–759. Springer, New York (2004)
7. Kumar, P., Yildirim, E.A.: Minimum-volume enclosing ellipsoids and core sets. J. Optim. Theory Appl. **126**, 1–21 (2005)
8. Khachiyan, L.G.: Rounding of polytopes in the real number model of computation. Math. Oper. Res. **21**, 307–320 (1996)
9. John, F.: Extremum problems with inequalities as subsidiary conditions. In: Studies and Essays, Presented to R. Courant on his 60th birthday, January 8, 1948, pp. 187–204. Interscience, New York (1948); Reprinted in: Fritz John, Collected Papers, vol. 2, pp. 543–560, edited by J. Moser, Birkhäuser, Boston (1985)
10. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Nav. Res. Logist. Q. **3**, 95–110 (1956)
11. Todd, M.J., Yildirim, E.A.: On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. Discrete Appl. Math. **155**, 1731–1744 (2007)
12. Yildirim, E.A.: On the minimum volume covering ellipsoid of ellipsoids. SIAM J. Optim. **17**, 621–641 (2006)
13. Kumar, P., Mitchell, J.S.B., Yildirim, E.A.: Approximate minimum enclosing balls in high dimensions using core-sets. ACM J. Exp. Algorithmics **8** (2003)