

Two Algorithms for the Minimum Enclosing Ball Problem

E. Alper Yıldırım*

May 3, 2007

Abstract

Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon > 0$, we propose and analyze two algorithms for the problem of computing a $(1 + \epsilon)$ -approximation to the radius of the minimum enclosing ball of \mathcal{A} . The first algorithm is closely related to the Frank-Wolfe algorithm with a proper initialization applied to the dual formulation of the minimum enclosing ball problem. We establish that this algorithm converges in $O(1/\epsilon)$ iterations with an overall complexity bound of $O(mn/\epsilon)$ arithmetic operations. In addition, the algorithm returns a “core set” of size $O(1/\epsilon)$, which is independent of both m and n . The latter algorithm is obtained by incorporating “away” steps into the former one at each iteration and achieves the same asymptotic complexity bound as the first one. While the asymptotic bound on the size of the core set returned by the second algorithm also remains the same as the first one, the latter algorithm has the potential to compute even smaller core sets in practice since, in contrast with the former one, it allows “dropping” points from the working core set at each iteration. Our computational results indicate that the latter algorithm indeed returns smaller core sets in comparison with the first one. We also discuss how our algorithms can be extended to compute an approximation to the minimum enclosing ball of other input sets. In particular, we establish the existence of a core set of size $O(1/\epsilon)$ for a much wider class of input sets.

Keywords: Minimum enclosing balls, core sets, approximation algorithms.

AMS Subject Classification: 90C25, 90C46, 65K05.

*Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey. e-mail: yildirim@bilkent.edu.tr

1 Introduction

Given a finite set of vectors $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$, we are concerned with the problem of computing an approximation to the minimum enclosing ball of \mathcal{A} , which we shall denote by $\text{MEB}(\mathcal{A})$.

For $c \in \mathbb{R}^n$ and a nonnegative $\rho \in \mathbb{R}$, let $\mathcal{B}_{c,\rho} \subset \mathbb{R}^n$ denote the ball centered at c with radius ρ , i.e.,

$$\mathcal{B}_{c,\rho} := \{x \in \mathbb{R}^n : \|x - c\| \leq \rho\},$$

where $\|\cdot\|$ denotes the Euclidean norm.

Given $\epsilon > 0$, a ball $\mathcal{B}_{c,\rho}$ is said to be a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ if

$$\mathcal{A} \subset \mathcal{B}_{c,\rho}, \quad \rho \leq (1 + \epsilon)\rho_{\mathcal{A}}, \tag{1}$$

where $\mathcal{B}_{c_{\mathcal{A}},\rho_{\mathcal{A}}} := \text{MEB}(\mathcal{A})$.

A subset $\mathcal{X} \subseteq \mathcal{A}$ is said to be an ϵ -core set (or a core set) of \mathcal{A} if

$$\rho_{\mathcal{X}} \leq \rho_{\mathcal{A}} \leq (1 + \epsilon)\rho_{\mathcal{X}}, \tag{2}$$

where $\mathcal{B}_{c_{\mathcal{X}},\rho_{\mathcal{X}}} := \text{MEB}(\mathcal{X})$. Small core sets play an important role in designing efficient algorithms for large-scale problems since they provide a compact representation of the input set \mathcal{A} . If an ϵ -core set \mathcal{X} is available, then solving the problem on \mathcal{X} already yields a good approximation to $\text{MEB}(\mathcal{A})$. Since the center $c_{\mathcal{A}}$ of $\text{MEB}(\mathcal{A})$ lies in the convex hull of \mathcal{A} (cf. Section 2), it follows from Carathéodory's theorem that there always exists a 0-core set of size at most $n + 1$.

Minimum enclosing balls have numerous important applications in clustering, nearest neighbor search, data classification, support vector machines, machine learning, facility location, collision detection, computer graphics, and military operations. We refer the reader to [21] and the references therein. In particular, many of these applications give rise to instances of the MEB problem with a large number of input points in moderately high dimensions and a reasonably small accuracy suffices for such applications.

The minimum enclosing ball problem has a fairly rich literature dating back to at least the 19th century [32]. One of the earliest known solution methods is given by Sylvester [33], which is attributed to Peirce, and later rediscovered by Chrystal [8]. The reader is referred to [5] for a detailed account of earlier history of this problem. More recent references include [22, 12, 24, 9, 7, 30, 6, 20, 19, 23, 27, 35, 14, 15, 4, 2, 21, 36, 11, 10, 25, 26, 18, 38, 28].

The earliest known algorithm due to Chrystal and Peirce [33, 8] computes the exact minimum enclosing ball of m points in the plane in $O(m^2)$ operations in the worst case. For a fixed dimension n , the minimum enclosing ball of m points can be computed in $O(m)$ operations [23, 35]. However, the dependence on the dimension n is exponential. Bădiou, Har-Peled, and Indyk [4] established the existence of an ϵ -core set of size $O(1/\epsilon^2)$. Note that the size of the core set is independent of m and n . Based on this result, their algorithm can compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn/\epsilon^2 + (1/\epsilon^{10}) \log(1/\epsilon))$ operations. Bădiou and Clarkson [2] and Kumar, Mitchell, and Yıldırım [21] independently

discovered the existence of an ϵ -core set of size $O(1/\epsilon)$. As noted in [2], this improved core set result can be combined with the algorithm of [4] to obtain an improved running time of $O(mn/\epsilon + 1/\epsilon^5)$. The algorithm of [21] achieves a slightly improved complexity bound of $O(mn/\epsilon + (1/\epsilon^{4.5}) \log(1/\epsilon))$ using second-order cone programming combined with column generation. In addition, Bădiou and Clarkson [2] proposed another simple algorithm that computes a $(1 + \epsilon)$ -approximation in $O(mn/\epsilon^2)$ operations. In another paper, the same authors established a tight upper bound of $\lceil 1/\epsilon \rceil$ on the size of an ϵ -core set [3]. However, their construction is based on the assumption that $n \geq \lceil 1/\epsilon \rceil$. The algorithm of Panigraphy [29] computes a $(1 + \epsilon)$ -approximation in $O(mn/\epsilon)$ operations. Note that this is the best known complexity bound for fixed ϵ . If ϵ is not fixed, the minimum enclosing ball can be formulated as an instance of convex programming problem and can be solved using the ellipsoid method in $O(n^3 m \log(1/\epsilon))$ operations [17]. Alternatively, interior-point methods yield an overall complexity bound of $O(n^2 m^{3/2} \log(1/\epsilon))$ operations if the problem is formulated as an instance of second-order cone programming [21].

In this paper, we focus on large-scale instances of the minimum enclosing ball problem for which a reasonably small value of ϵ is satisfactory. We propose two algorithms that compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ for a given $\epsilon > 0$. Our first algorithm is closely related to the Frank-Wolfe algorithm [13] applied to the dual formulation of the problem. The second algorithm is obtained by incorporating “away” steps into each iteration of the first one. As such, the latter algorithm has the potential to compute a smaller core set than the former one since it allows “dropping” points from the working core set at each iteration. A similar algorithm has recently been proposed for the minimum-volume enclosing ellipsoid problem [34]. Both of our algorithms compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn/\epsilon)$ operations, which matches the currently best known complexity bound for fixed ϵ . In addition, each algorithm explicitly computes a core set of size $O(1/\epsilon)$. Our computational results indicate that the sizes of the core sets returned by our algorithms are generally much smaller than the corresponding worst-case estimates. Furthermore, as expected, the latter algorithm almost always computes a smaller core set than the former one.

We also discuss how our algorithms can be extended to compute an approximate minimum enclosing ball of more general input sets. In particular, we establish that the asymptotic core set size of $O(1/\epsilon)$ extends to a much larger class of input sets.

This paper is organized as follows. In the remainder of this section, we define our notation. In Section 2, we discuss optimization formulations for the minimum enclosing ball problem. Section 3 presents our first algorithm. The second algorithm is the topic of Section 4. We discuss the extensions of our algorithms in Section 5. The computational results are presented in Section 6. Finally, we conclude the paper with some future research directions in Section 7.

1.1 Notation

Vectors are denoted by lower-case Roman letters. For a vector p , p_i denotes its i th component. Inequalities on vectors apply to each component. We reserve e^j for the j th unit vector. Upper-case Roman letters are reserved for matrices. We use $\log(\cdot)$ to denote

the natural logarithm. Functions and operators are denoted by upper-case Greek letters. Scalars except for m and n are represented by lower-case Greek letters unless they represent components of a vector or elements of a sequence of scalars, vectors, or matrices. We reserve i, j , and k for such indexing purposes. Upper-case script letters are used for all other objects such as sets, balls, and ellipsoids.

2 Optimization Formulations

Let $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$. The minimum enclosing ball of \mathcal{A} can be computed by solving the following optimization problem:

$$\begin{aligned}
 (\mathcal{P}_1) \quad & \min_{c, \rho} \quad \rho \\
 & \text{subject to} \\
 & \quad \|a^i - c\| \leq \rho, \quad i = 1, \dots, m,
 \end{aligned}$$

where $c \in \mathbb{R}^n$ and $\rho \in \mathbb{R}$ are the decision variables. By squaring the constraints and defining $\gamma := \rho^2$, (\mathcal{P}_1) can be converted into the following optimization problem with smooth, convex quadratic constraints:

$$\begin{aligned}
 (\mathcal{P}_2) \quad & \min_{c, \gamma} \quad \gamma \\
 & \text{subject to} \\
 & \quad (a^i)^T a^i - 2(a^i)^T c + c^T c \leq \gamma, \quad i = 1, \dots, m.
 \end{aligned}$$

The Lagrangian dual of (\mathcal{P}_2) is given by

$$\begin{aligned}
 (\mathcal{D}) \quad & \max_u \quad \Phi(u) := \sum_{i=1}^m u_i (a^i)^T a^i - \left(\sum_{i=1}^m u_i a^i \right)^T \left(\sum_{i=1}^m u_i a^i \right) \\
 & \text{subject to} \\
 & \quad \sum_{i=1}^m u_i = 1, \\
 & \quad u_i \geq 0, \quad i = 1, \dots, m,
 \end{aligned}$$

where $u \in \mathbb{R}^m$ is the decision variable.

$(c_{\mathcal{A}}, \gamma_{\mathcal{A}}) \in \mathbb{R}^n \times \mathbb{R}$ is an optimal solution of (\mathcal{P}_2) if and only if there exists $u^* \in \mathbb{R}^m$ such that

$$\sum_{i=1}^m u_i^* = 1, \tag{3a}$$

$$c_{\mathcal{A}} = \sum_{i=1}^m u_i^* a^i, \tag{3b}$$

$$(a^i)^T a^i - 2(a^i)^T c_{\mathcal{A}} + (c_{\mathcal{A}})^T c_{\mathcal{A}} \leq \gamma_{\mathcal{A}}, \quad i = 1, \dots, m, \tag{3c}$$

$$u_i^* ((a^i)^T a^i - 2(a^i)^T c_{\mathcal{A}} + (c_{\mathcal{A}})^T c_{\mathcal{A}} - \gamma_{\mathcal{A}}) = 0, \quad i = 1, \dots, m, \tag{3d}$$

along with $u^* \geq 0$. A simple manipulation of the optimality conditions reveals that

$$\gamma_{\mathcal{A}} = \Phi(u^*), \quad (4)$$

which implies that $u^* \in \mathbb{R}^m$ is an optimal solution of (\mathcal{D}) and that strong duality holds between (\mathcal{P}_2) and (\mathcal{D}) . Note that the center $c_{\mathcal{A}}$ of the minimum enclosing ball of \mathcal{A} is given by a convex combination of the elements of \mathcal{A} by (3b). In addition, it follows from (3d) that only the components of u^* corresponding to the points on the boundary of $\text{MEB}(\mathcal{A})$ can have a positive value.

Lemma 2.1 *Let $\mathcal{A} = \{a^1, \dots, a^m\}$. The minimum enclosing ball of \mathcal{A} exists and is unique. Let $u^* \in \mathbb{R}^m$ denote the optimal solution of (\mathcal{D}) . Then, $\text{MEB}(\mathcal{A}) = \mathcal{B}_{c_{\mathcal{A}}, \rho_{\mathcal{A}}}$, where*

$$c_{\mathcal{A}} = \sum_{i=1}^m u_i^* a^i, \quad \rho_{\mathcal{A}} = \sqrt{\Phi(u^*)}. \quad (5)$$

Proof. Note that $\mathcal{A} \subset \mathcal{B}_{0, \rho^u}$, where $\rho^u := \max_{i=1, \dots, m} \|a^i\|$. By adding the redundant constraint $\gamma \leq (\rho^u)^2$ to (\mathcal{P}_2) , the feasible region becomes a closed and bounded set and the objective function is continuous, which establishes the existence of $\text{MEB}(\mathcal{A})$. If there were two different minimum enclosing balls, one can then construct a ball of smaller radius that encloses the intersection of the two balls and hence also \mathcal{A} , which is a contradiction. The relationships (5) directly follow from the discussions preceding the lemma. \square

By Lemma 2.1, $\text{MEB}(\mathcal{A})$ can be computed by solving the dual problem (\mathcal{D}) , which will be the basis of both of our algorithms in this paper. We close this section by the following technical result, which will play an important role in finding a good initial feasible solution in our algorithms. The reader is referred to [16] for the proof of this result.

Lemma 2.2 *Let $\mathcal{A} = \{a^1, \dots, a^m\}$ and let $\text{MEB}(\mathcal{A}) = \mathcal{B}_{c_{\mathcal{A}}, \rho_{\mathcal{A}}}$. Then, any closed halfspace that contains $c_{\mathcal{A}}$ also contains at least one point $a^j \in \mathcal{A}$ such that $\|a^j - c_{\mathcal{A}}\| = \rho_{\mathcal{A}}$.*

3 The First Algorithm

Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon > 0$, we present our first algorithm that computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in this section.

Algorithm 3.1 The first algorithm that computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$.

Require: Input set of points $\mathcal{A} = \{a^1, \dots, a^m\} \subset \mathbb{R}^n, \epsilon > 0$.

- 1: $\alpha \leftarrow \arg \max_{i=1, \dots, m} \|a^i - a^1\|^2, \quad \beta \leftarrow \arg \max_{i=1, \dots, m} \|a^i - a^\alpha\|^2;$
 - 2: $u_i^0 \leftarrow 0, \quad i = 1, \dots, m;$
 - 3: $u_\alpha^0 \leftarrow 1/2, \quad u_\beta^0 \leftarrow 1/2;$
 - 4: $\mathcal{X}_0 \leftarrow \{a^\alpha, a^\beta\};$
 - 5: $c^0 \leftarrow \sum_{i=1}^m u_i^0 a^i;$
 - 6: $\gamma^0 \leftarrow \Psi(u^0);$
 - 7: $\kappa \leftarrow \arg \max_{i=1, \dots, m} \|a^i - c^0\|^2;$
 - 8: $\delta_0 \leftarrow \left(\|a^\kappa - c^0\|^2 / \gamma^0 \right) - 1;$
 - 9: $k \leftarrow 0;$
 - 10: **While** $\delta_k > (1 + \epsilon)^2 - 1$, **do**
 - 11: **loop**
 - 12: $\lambda^k \leftarrow \delta_k / [2(1 + \delta_k)];$
 - 13: $k \leftarrow k + 1;$
 - 14: $u^k \leftarrow (1 - \lambda^{k-1})u^{k-1} + \lambda^{k-1}e^\kappa;$
 - 15: $c^k \leftarrow (1 - \lambda^{k-1})c^{k-1} + \lambda^{k-1}a^\kappa;$
 - 16: $\mathcal{X}_k \leftarrow \mathcal{X}_{k-1} \cup \{a^\kappa\};$
 - 17: $\gamma^k \leftarrow \Psi(u^k);$
 - 18: $\kappa \leftarrow \arg \max_{i=1, \dots, m} \|a^i - c^k\|^2;$
 - 19: $\delta_k \leftarrow \left(\|a^\kappa - c^k\|^2 / \gamma^k \right) - 1;$
 - 20: **end loop**
 - 21: **Output** $c^k, \mathcal{X}_k, u^k, \sqrt{(1 + \delta_k)\gamma^k}$.
-

We now describe Algorithm 3.1 in more detail. In Step 1, the algorithm computes the furthest point $a^\alpha \in \mathcal{A}$ from $a^1 \in \mathcal{A}$ and then computes the furthest point $a^\beta \in \mathcal{A}$ from a^α . Steps 2 and 3 initialize the vector $u^0 \in \mathbb{R}^m$. Note that u^0 is a feasible solution of the dual problem (\mathcal{D}) . The core set \mathcal{X}_0 is initialized at Step 4. At each iteration, the algorithm implicitly constructs a “trial” ball with center c^k and radius $(\gamma^k)^{1/2}$. By Lemma 2.1, this ball coincides with $\text{MEB}(\mathcal{A})$ if and only if u^k is an optimal solution of (\mathcal{D}) . Otherwise, at least one point in \mathcal{A} lies outside of this ball. Note that δ_k satisfies $\|a^\kappa - c^k\|^2 = (1 + \delta_k)\gamma^k$, where $a^\kappa \in \mathcal{A}$ is the furthest point from c^k . It follows that the trial ball encloses \mathcal{A} if its radius is expanded by a factor of $(1 + \delta_k)^{1/2}$. Unless the termination criterion is satisfied, the new center c^{k+1} is computed by shifting c^k towards the furthest point a^κ , which is added to the working core set \mathcal{X}_{k+1} , and u^{k+1} is updated accordingly to ensure that dual feasibility is maintained. The algorithm continues in an iterative manner by computing a new trial ball corresponding to u^{k+1} .

Algorithm 3.1 is the adaptation of the Frank-Wolfe algorithm to the dual problem (\mathcal{D}) . At each iteration, the quadratic objective function $\Phi(u)$ of (\mathcal{D}) is linearized at the current feasible solution u^k . Since the feasible region of (\mathcal{D}) is the unit simplex, the unit vector e^κ , where κ is the index of the furthest point in \mathcal{A} from c_k , solves the linearized subproblem. It

is easy to verify that

$$\lambda_k = \arg \max_{\lambda \in [0,1]} \Phi((1-\lambda)u^k + \lambda e^\kappa).$$

We remark that Algorithm 3.1 uses only the first-order approximation. As such, each iteration is fairly cheap but the number of iterations is usually significantly higher than other algorithms that use second-order information such as interior-point methods. However, such general-purpose algorithms become computationally infeasible for larger problems since each iteration is usually much more expensive. This observation provides one of our motivations to develop a specialized algorithm for this problem.

3.1 Analysis of the First Algorithm

This subsection is devoted to the analysis of Algorithm 3.1.

Lemma 3.1 $u^0 \in \mathbb{R}^m$ satisfies $\gamma^0 = \Psi(u^0) \geq (1/3)\Psi(u^*) = (1/3)\gamma_{\mathcal{A}}$, where $u^* \in \mathbb{R}^m$ and $\gamma_{\mathcal{A}}$ are the optimal solution and the optimal value of (\mathcal{D}) , respectively. Furthermore, $\delta_0 \leq 8$.

Proof. Note that

$$\Psi(u^0) = (1/2)(a^\alpha)^T a^\alpha + (1/2)(a^\beta)^T a^\beta - (c^0)^T c^0, \quad (6a)$$

$$= (1/4)(a^\alpha)^T a^\alpha + (1/4)(a^\beta)^T a^\beta - (1/2)(a^\alpha)^T a^\beta, \quad (6b)$$

$$= (1/4) \|a^\alpha - a^\beta\|^2, \quad (6c)$$

where we used the fact that $c^0 = (1/2)(a^\alpha + a^\beta)$. The proof is based on establishing that at least one of a^α and a^β is sufficiently away from the center $c_{\mathcal{A}}$ of $\text{MEB}(\mathcal{A})$.

First, suppose that $\|a^1 - c_{\mathcal{A}}\| \geq (1/\sqrt{3})\rho_{\mathcal{A}}$, where $\rho_{\mathcal{A}}$ is the radius of $\text{MEB}(\mathcal{A})$. Let H be the hyperplane passing through $c_{\mathcal{A}}$ that is perpendicular to $a^1 - c_{\mathcal{A}}$. Let H_+ denote the closed halfspace whose boundary is H and which does not contain a^1 . By Lemma 2.2, H_+ contains a point $a^j \in \mathcal{A}$ such that $\|a^j - c_{\mathcal{A}}\| = \rho_{\mathcal{A}}$. Therefore, $\|a^\alpha - a^1\|^2 \geq \|a^1 - c_{\mathcal{A}}\|^2 + (\rho_{\mathcal{A}})^2 \geq (4/3)\gamma_{\mathcal{A}}$, where $\gamma_{\mathcal{A}} = \Psi(u^*) = (\rho_{\mathcal{A}})^2$ is the optimal value of (\mathcal{D}) . It follows from (6) that

$$\Psi(u^0) = (1/4) \|a^\beta - a^\alpha\|^2 \geq (1/4) \|a^1 - a^\alpha\|^2 \geq (1/3)\Psi(u^*).$$

Suppose now that $\|a^1 - c_{\mathcal{A}}\| = \theta\rho_{\mathcal{A}}$, where $\theta < 1/\sqrt{3}$. In this case, $\|a^1 - a^\alpha\| \leq \|a^1 - c_{\mathcal{A}}\| + \|c_{\mathcal{A}} - a^\alpha\|$, which implies that

$$\|c_{\mathcal{A}} - a^\alpha\| \geq \|a^1 - a^\alpha\| - \|a^1 - c_{\mathcal{A}}\| \geq (1 + \theta^2)^{1/2}\rho_{\mathcal{A}} - \theta\rho_{\mathcal{A}} = [(1 + \theta^2)^{1/2} - \theta]\rho_{\mathcal{A}},$$

where we again invoked Lemma 2.2 to obtain a lower bound on $\|a^1 - a^\alpha\|$. Therefore, one more application of Lemma 2.2 yields

$$\begin{aligned} \Psi(u^0) &= (1/4) \|a^\beta - a^\alpha\|^2, \\ &\geq (1/4) (\|a^\alpha - c_{\mathcal{A}}\|^2 + (\rho_{\mathcal{A}})^2), \\ &\geq (1/4) (1 + \theta^2 + \theta^2 - 2\theta(1 + \theta^2)^{1/2} + 1) \gamma_{\mathcal{A}}, \\ &= (1/2) (1 + \theta^2 - \theta(1 + \theta^2)^{1/2}) \gamma_{\mathcal{A}}. \end{aligned}$$

It is easy to verify that $(1/2)(1 + \theta^2 - \theta(1 + \theta^2)^{1/2})$ is a decreasing function of θ . Since $\theta < 1/\sqrt{3}$, it follows that

$$\Psi(u^0) \geq (1/2)(1 + 1/3 - 2/3)\gamma_{\mathcal{A}} = (1/3)\Psi(u^*),$$

which completes the first part of the proof.

Let a^κ be the furthest point in \mathcal{A} from c^0 . Then,

$$\|a^\kappa - c^0\| \leq \|a^\kappa - a^\alpha\| + \|a^\alpha - c^0\| \leq \|a^\beta - a^\alpha\| + (1/2)\|a^\beta - a^\alpha\| = (3/2)\|a^\beta - a^\alpha\|,$$

where we used the definition of c^0 and the fact that a^β is the furthest point in \mathcal{A} from a^α . Therefore, $\delta_0 = (\|a^\kappa - c^0\|^2/\gamma^0) - 1 \leq [4(9/4)(\gamma^0/\gamma^0)] - 1 = 8$, where we used (6). The second part of the assertion follows. \square

Lemma 3.1 establishes several properties of the initial feasible solution $u^0 \in \mathbb{R}^m$. The next lemma relates the dual objective function values evaluated at the successive iterates generated by Algorithm 3.1.

Lemma 3.2 *For each $k = 0, 1, \dots$, the following relationship is satisfied:*

$$\gamma^{k+1} = \gamma^k \left(1 + \frac{\delta_k^2}{4(1 + \delta_k)} \right). \quad (7)$$

Proof. Let a^κ denote the furthest point from c^k . Then, $u^{k+1} = (1 - \lambda^k)u^k + \lambda^k e^\kappa$. Therefore,

$$\begin{aligned} \gamma^{k+1} &= \Psi((1 - \lambda^k)u^k + \lambda^k e^\kappa), \\ &= \sum_{i=1}^m (1 - \lambda^k)u_i^k (a^i)^T (a^i) + \lambda^k (a^\kappa)^T (a^\kappa) - \left\| \sum_{i=1}^m (1 - \lambda^k)u_i^k (a^i) + \lambda^k (a^\kappa) \right\|^2, \\ &= (1 - \lambda^k)\gamma^k + \lambda^k (1 - \lambda^k) \|a^\kappa - c^k\|^2, \\ &= (1 - \lambda^k)\gamma^k + \lambda^k (1 - \lambda^k)(1 + \delta_k)\gamma^k, \\ &= \gamma^k \left(1 + \frac{\delta_k^2}{4(1 + \delta_k)} \right), \end{aligned}$$

where we used the definition of δ_k in the fourth equality and the definition of λ^k in the last equality. \square

We now focus on establishing an upper bound on the number of iterations required to have an iterate u^k with δ_k sufficiently small. To that end, let us define

$$\tau_\nu := \min \left\{ k : \delta_k \leq \frac{1}{2^\nu} \right\}, \quad \nu = 0, 1, \dots \quad (8)$$

Lemma 3.3 τ_ν satisfies the following relationships:

$$\tau_0 \leq 9, \quad (9a)$$

$$\tau_\nu - \tau_{\nu-1} \leq 17(2^\nu) \quad \nu = 1, 2, \dots \quad (9b)$$

Proof. Let us first consider τ_0 . At each iteration $k < \tau_0$, we have $\delta_k > 1$. By Lemma 3.2,

$$\begin{aligned}\gamma^{k+1} &= \gamma^k \left(1 + \frac{\delta_k^2}{4(1 + \delta_k)} \right), \\ &\geq \gamma^k (1 + 1/8).\end{aligned}$$

Iterating this inequality, we obtain $\gamma^{k+1} \geq (9/8)^{k+1} \gamma^0$. By Lemma 3.1 and the feasibility of u^{k+1} , we have

$$\gamma_{\mathcal{A}} \geq \gamma^{k+1} \geq (9/8)^{k+1} \gamma^0 \geq (9/8)^{k+1} (\gamma_{\mathcal{A}}/3),$$

which implies that $k + 1 \leq \log(3)/\log(9/8) \leq 10$, or equivalently that $k \leq 9$. Therefore, $\tau_0 \leq 9$.

Let us now consider $\tau_\nu - \tau_{\nu-1}$ for $\nu = 1, 2, \dots$. Let $\mu := \tau_{\nu-1}$. At each iteration k with $\delta^k > 1/2^\nu$, we similarly have

$$\begin{aligned}\gamma^{k+1} &= \gamma^k \left(1 + \frac{\delta_k^2}{4(1 + \delta_k)} \right), \\ &\geq \gamma^k \left(1 + \frac{1}{2^{2+\nu}(2^\nu + 1)} \right).\end{aligned}$$

At iteration μ , we have $\delta_\mu \leq 1/2^{\nu-1}$. Since the ball centered at c_μ with radius $[(1 + \delta_\mu)\gamma^\mu]^{1/2}$ encloses \mathcal{A} , it follows that $\gamma^\mu \leq \gamma_{\mathcal{A}} \leq (1 + \delta_\mu)\gamma^\mu \leq (1 + (1/2^{\nu-1}))\gamma^\mu$. Together with the repeated application of the inequality above, we have

$$\gamma_{\mathcal{A}} \geq \gamma^{\mu+k} \geq \gamma^\mu \left(1 + \frac{1}{2^{2+\nu}(2^\nu + 1)} \right)^k \geq \frac{\gamma_{\mathcal{A}}}{1 + (1/2^{\nu-1})} \left(1 + \frac{1}{2^{2+\nu}(2^\nu + 1)} \right)^k,$$

which implies that

$$\tau_\nu - \tau_{\nu-1} \leq \frac{\log\left(1 + \frac{1}{2^{\nu-1}}\right)}{\log\left(1 + \frac{1}{2^{2+\nu}(2^\nu + 1)}\right)} \leq \frac{1}{2^{\nu-1}} \frac{\frac{1}{2^{2+\nu}(2^\nu + 1)} + 1}{\frac{1}{2^{2+\nu}(2^\nu + 1)}} \leq 8 \left(\frac{25}{24}\right) (2^\nu + 1) \leq 17(2^\nu),$$

where we used the inequalities $\log(1 + x) \leq x$ for $x > -1$ and $\log(1 + x) \geq x/(x + 1)$ for $x > -1$. \square

The following lemma establishes an upper bound on the number of iterations to obtain an iterate with $\delta_k \leq \delta$.

Lemma 3.4 *Let $\delta \in (0, 1)$. Algorithm 3.1 computes an iterate k with $\delta_k \leq \delta$ in at most $9 + 68/\delta$ iterations.*

Proof. Let σ be an integer such that $1/2^\sigma \leq \delta \leq 2/2^\sigma$. Therefore, after at most τ_σ iterations, Algorithm 3.1 computes an iterate k with $\delta_k \leq \delta$. By Lemma 3.3,

$$\tau_\sigma = \tau_0 + \sum_{\nu=1}^{\sigma} (\tau_\nu - \tau_{\nu-1}) \leq 9 + 17 \sum_{\nu=1}^{\sigma} 2^\nu \leq 9 + 34(2^\sigma) \leq 9 + 68/\delta.$$

\square

We now have all the ingredients to establish the iteration complexity of Algorithm 3.1.

Theorem 3.1 *Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon \in (0, 1)$, Algorithm 3.1 computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in at most $9 + 34/\epsilon$ iterations.*

Proof. Let u^η denote the final iterate computed by Algorithm 3.1 and let $\gamma^\eta = \Psi(u^\eta)$. Then, the trial ball centered at c^η with radius $[(1 + \delta_\eta)\gamma^\eta]^{1/2}$ encloses \mathcal{A} . Note that u^η is a feasible solution of (\mathcal{D}) and $\delta_\eta \leq (1 + \epsilon)^2 - 1$ by the termination criterion. Therefore, $(\gamma^\eta)^{1/2} \leq \rho_{\mathcal{A}} \leq [(1 + \delta_\eta)\gamma^\eta]^{1/2} \leq (1 + \epsilon)(\gamma^\eta)^{1/2}$, which implies that the ball centered at c_η with radius $[(1 + \delta_\eta)\gamma^\eta]^{1/2}$ is a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$.

By Lemma 3.4, Algorithm 3.1 computes such an iterate with $\delta \leq (1 + \epsilon)^2 - 1$ in at most $9 + 68/(2\epsilon + \epsilon^2) \leq 9 + 34/\epsilon$ iterations. \square

Theorem 3.1 establishes that Algorithm 3.1 converges in $O(1/\epsilon)$ iterations. The next result presents the overall complexity.

Theorem 3.2 *Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon \in (0, 1)$, Algorithm 3.1 computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in at most $O(mn/\epsilon)$ arithmetic operations.*

Proof. The computation of the initial feasible solution u^0 requires two furthest point computations, which can be performed in $O(mn)$ operations. At each iteration, the dominating work is the computation of the furthest point from the center of the current trial ball, which also requires $O(mn)$ operations (note that γ^k can be updated using (7) in $O(1)$ operations). The result follows from Theorem 3.1. \square

We remark that the overall complexity of Algorithm 3.1 is linear in the number of points m and also linear in the dimension n . As such, the worst-case running time asymptotically matches the currently best known bound due to [29]. In particular, Theorem 3.2 suggests that Algorithm 3.1 is especially well-suited for large instances of the minimum enclosing ball problem where a moderately small value of ϵ (such as 10^{-3}) would be satisfactory.

We close this section by establishing that Algorithm 3.1 explicitly computes a core set of size $O(1/\epsilon)$, which also asymptotically matches the currently best known bound.

Theorem 3.3 *Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon \in (0, 1)$, let η denote the index of the final iterate computed by Algorithm 3.1. Then, $\mathcal{X}_\eta \subseteq \mathcal{A}$ is an ϵ -core set of \mathcal{A} . Furthermore, $|\mathcal{X}_\eta| = O(1/\epsilon)$.*

Proof. Let u^η denote the final iterate returned by Algorithm 3.1 and let $\gamma^\eta = \Psi(u^\eta)$. Clearly, the restriction of u^η to its positive entries is a feasible solution of the dual formulation of the minimum enclosing ball problem for \mathcal{X}_η . Therefore, $\gamma^\eta \leq (\rho_{\mathcal{X}_\eta})^2 \leq (\rho_{\mathcal{A}})^2$, where $\rho_{\mathcal{X}_\eta}$ is the radius of $\text{MEB}(\mathcal{X}_\eta)$. However, $\gamma_{\mathcal{A}} = (\rho_{\mathcal{A}})^2 \leq (1 + \delta_\eta)\gamma^\eta \leq (1 + \epsilon)^2\gamma^\eta$ by Theorem 3.1. Combining these inequalities, we obtain $\rho_{\mathcal{X}_\eta} \leq \rho_{\mathcal{A}} \leq (1 + \epsilon)(\gamma^\eta)^{1/2} \leq (1 + \epsilon)\rho_{\mathcal{X}_\eta}$ as desired.

Note that $|\mathcal{X}_\eta|$ is precisely equal to the number of positive components of u^η . However, the initial solution u^0 has only two positive components. Each iteration can add at most one positive component to u^k . Therefore, $|\mathcal{X}_\eta| = O(1/\epsilon)$ by Theorem 3.1. \square

4 The Second Algorithm

In this section, we describe our second algorithm, which is a modification of Algorithm 3.1.

Algorithm 4.1 starts off with the same initial solution u^0 as the one computed by Algorithm 3.1. At each iteration, the furthest point in \mathcal{A} from the center c^k of the trial ball is computed as in Algorithm 3.1. In contrast, each iteration of Algorithm 4.1 also includes the computation of the *closest* point to c^k among all points in $\mathcal{X}_k \subseteq \mathcal{A}$. Geometrically, this is equivalent to shrinking the current trial ball by the smallest amount so that the shrunken ball does not contain any points in \mathcal{X}_k . Algebraically, this corresponds to moving away from the vertex of the unit simplex that minimizes the linear approximation to $\Psi(u)$ at u^k , where the minimization is over the vertices $\{e^j : u_j^k > 0\}$. The feasible solution u^k is updated in different ways based on these two computations. If $\delta_k = \delta_k^+$, then Algorithm 4.1 uses the exact same update as in Algorithm 3.1. Otherwise, the new center c^{k+1} is obtained by moving the current center c^k *away* from the closest point $a^\xi \in \mathcal{X}_k$. Therefore, Algorithm 4.1 is obtained by incorporating “away” steps into Algorithm 3.1. For away steps, it is easy to verify that

$$\lambda^k = \arg \max_{\lambda \in [0, u_\xi^k / (1 - u_\xi^k)]} \Psi((1 + \lambda)u^k - \lambda e^\xi). \quad (10)$$

Note that the range of λ is chosen to ensure that the dual feasibility constraint $u^{k+1} \geq 0$ is satisfied.

4.1 Analysis of the Second Algorithm

The analysis of Algorithm 4.1 is very similar to that of Algorithm 3.1. As in [34], we call iteration k a *plus*-iteration if $\delta_k = \delta_k^+$. If $\delta_k = \delta_k^-$ and $\lambda^k = (\delta_k^-) / [2(1 - \delta_k^-)]$, then we call it a *minus*-iteration. The working core set remains unchanged at a minus-iteration. Finally, if $\delta_k = \delta_k^-$ and $\lambda^k = u_\xi^k / (1 - u_\xi^k)$, we then call it a *drop*-iteration since the ξ th component of u^k drops to 0 and a^ξ is removed from the working core set.

Our analysis mimics the analysis of [34] for a similar algorithm that computes an approximation to the minimum-volume enclosing ellipsoid of a finite set of points. The next lemma establishes a lower bound on the improvement at each plus- or minus-iteration.

Lemma 4.1 *At each plus- or minus-iteration,*

$$\gamma^{k+1} \geq \left(1 + \frac{\delta_k^2}{4(1 + \delta_k)}\right), \quad k = 0, 1, \dots \quad (11)$$

Proof. At a plus-iteration, the results directly follows from Lemma 3.2. At a minus-iteration,

$$\begin{aligned} \gamma^{k+1} &= \Psi((1 + \lambda_k)u^k - \lambda^k e^\xi), \\ &= \gamma^k \left(1 + \frac{(\delta_k^-)^2}{4(1 - \delta_k^-)}\right). \end{aligned}$$

Algorithm 4.1 The second algorithm that computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$.

Require: Input set of points $\mathcal{A} = \{a^1, \dots, a^m\} \subset \mathbb{R}^n, \epsilon > 0$.

- 1: $\alpha \leftarrow \arg \max_{i=1, \dots, m} \|a^i - a^1\|^2, \quad \beta \leftarrow \arg \max_{i=1, \dots, m} \|a^i - a^\alpha\|^2;$
 - 2: $u_i^0 \leftarrow 0, \quad i = 1, \dots, m;$
 - 3: $u_\alpha^0 \leftarrow 1/2, \quad u_\beta^0 \leftarrow 1/2;$
 - 4: $\mathcal{X}_0 \leftarrow \{a^\alpha, a^\beta\};$
 - 5: $c^0 \leftarrow \sum_{i=1}^m u_i^0 a^i;$
 - 6: $\gamma^0 \leftarrow \Psi(u^0);$
 - 7: $\kappa \leftarrow \arg \max_{i=1, \dots, m} \|a^i - c^0\|^2, \quad \xi \leftarrow \arg \min_{i: a^i \in \mathcal{X}_0} \|a^i - c^0\|^2;$
 - 8: $\delta_0^+ \leftarrow \left(\|a^\kappa - c^0\|^2 / \gamma^0 \right) - 1, \quad \delta_0^- \leftarrow 1 - \left(\|a^\xi - c^0\|^2 / \gamma^0 \right);$
 - 9: $\delta_0 \leftarrow \max\{\delta_0^+, \delta_0^-\};$
 - 10: $k \leftarrow 0;$
 - 11: While $\delta_k > (1 + \epsilon)^2 - 1$, do
 - 12: **loop**
 - 13: **if** $\delta_k > \delta_k^-$ **then**
 - 14: $\lambda^k \leftarrow \delta_k / [2(1 + \delta_k)];$
 - 15: $k \leftarrow k + 1;$
 - 16: $u^k \leftarrow (1 - \lambda^{k-1})u^{k-1} + \lambda^{k-1}e^\kappa;$
 - 17: $c^k \leftarrow (1 - \lambda^{k-1})c^{k-1} + \lambda^{k-1}a^\kappa;$
 - 18: $\mathcal{X}_k \leftarrow \mathcal{X}_{k-1} \cup \{a^\kappa\};$
 - 19: **else**
 - 20: $\lambda^k \leftarrow \min \left\{ \frac{\delta_k^-}{2(1 - \delta_k^-)}, \frac{u_\xi^k}{1 - u_\xi^k} \right\};$
 - 21: $k \leftarrow k + 1;$
 - 22: **if** $\lambda^k = u_\xi^k / (1 - u_\xi^k)$ **then**
 - 23: $\mathcal{X}_k \leftarrow \mathcal{X}_{k-1} \setminus \{a^\xi\};$
 - 24: **else**
 - 25: $\mathcal{X}_k \leftarrow \mathcal{X}_{k-1};$
 - 26: **end if**
 - 27: $u^k \leftarrow (1 + \lambda^{k-1})u^{k-1} - \lambda^{k-1}e^\xi;$
 - 28: $c^k \leftarrow (1 + \lambda^{k-1})c^{k-1} - \lambda^{k-1}a^\xi;$
 - 29: **end if**
 - 30: $\gamma^k \leftarrow \Psi(u^k);$
 - 31: $\kappa \leftarrow \arg \max_{i=1, \dots, m} \|a^i - c^k\|^2, \quad \xi \leftarrow \arg \min_{i: a^i \in \mathcal{X}_k} \|a^i - c^k\|^2;$
 - 32: $\delta_k^+ \leftarrow \left(\|a^\kappa - c^k\|^2 / \gamma^k \right) - 1, \quad \delta_k^- \leftarrow 1 - \left(\|a^\xi - c^k\|^2 / \gamma^k \right);$
 - 33: $\delta_k \leftarrow \max\{\delta_k^+, \delta_k^-\};$
 - 34: **end loop**
 - 35: **Output** $c^k, \mathcal{X}_k, u^k, \sqrt{(1 + \delta_k)\gamma^k}$.
-

The result easily follows from the observation that

$$\frac{(\delta_k^-)^2}{4(1 - \delta_k^-)} \geq \frac{(\delta_k^-)^2}{4(1 + \delta_k^-)},$$

and that $\delta_k^- = \delta^k$ at a minus-iteration. □

Lemma 4.1 establishes that Algorithm 4.1 makes at least as much improvement as Algorithm 3.1 at each plus- or minus-iteration. At a drop-iteration, it is easy to show that $\gamma^{k+1} \geq \gamma^k$. However, we can no longer find a positive lower bound on $\gamma^{k+1} - \gamma^k \geq 0$. Using a similar reasoning as in [34], each drop-iteration can be paired with a previous plus-iteration k at which u_ξ^k was increased from 0, except for the α th and β th components which were positive at the initial solution and may be decreased to zero for the first time. Therefore, we can double the iteration count (and add two iterations to account for the initial positive components of u^0) in the analysis of Algorithm 3.1 to establish that Algorithm 4.1 can compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in at most twice as many iterations as that required by Algorithm 3.1. Note that this does not affect the asymptotic iteration bound of Algorithm 3.1. Furthermore, each iteration still requires $O(mn)$ operations, which implies that the asymptotic overall complexity of Algorithm 4.1 also remains the same as that of Algorithm 3.1. Finally, the asymptotic bound on the size of the core set is also unaffected. However, we remark that Algorithm 4.1 has the potential to compute even smaller core sets than those returned by Algorithm 3.1 due to the possible inclusion of minus- and drop-iterations. We summarize these results in the following theorem.

Theorem 4.1 *Given $\mathcal{A} := \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ and $\epsilon \in (0, 1)$, Algorithm 4.1 computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn/\epsilon)$ operations. Furthermore, upon termination, $\mathcal{X}_\eta \subseteq \mathcal{A}$ is an ϵ -core set and $|\mathcal{X}_\eta| = O(1/\epsilon)$, where η is the index of the final iterate computed by Algorithm 4.1.*

5 Extensions

In this section, we establish that the algorithmic frameworks of Sections 3 and 4 can be used to compute an approximation to the minimum enclosing ball of more general input sets. While the cost of each iteration of the corresponding algorithms may depend on the input set, the iteration complexity and the asymptotic size of the core set remain unchanged. Therefore, the existence of an ϵ -core set of size $O(1/\epsilon)$ extends to more general sets including those with uncountably many elements.

Note that both of the algorithms in this paper compute the initial feasible solution in a similar fashion. This computation entails finding the furthest point in the input set from a fixed point. In addition, similar furthest point computations are performed at each iteration of both of the algorithms. Therefore, the extent of input sets which are amenable to these algorithms highly depends on the efficiency with which such computations can be performed. In addition, each iteration of Algorithm 4.1 requires the computation of the closest point

in \mathcal{X}_k from a fixed point. Since \mathcal{X}_k is always a finite set, this computation can always be efficiently performed. For input sets with infinitely many points, note that the dimension of the vectors u^k generated by either of the two algorithms is no longer fixed and may increase by one at each iteration.

Another key observation is that Lemma 3.1, which establishes the quality of the initial feasible solution computed by each of the two algorithms, extends to more general sets. This follows from the fact that Lemma 2.2 remains true for arbitrary input sets. The proof of Lemma 2.2 is based on the argument that an enclosing ball of smaller radius can be constructed by moving the center away from the halfspace in the direction of the normal vector of the bounding hyperplane if the hypothesis of Lemma 2.2 is not satisfied by that halfspace. Therefore, the quality of the initial solution is independent of the input set.

We now specify several input sets for which similar algorithmic frameworks can be applied.

5.1 Set of Balls

Let $\mathcal{A} = \{\mathcal{B}_1, \dots, \mathcal{B}_m\} \subset \mathbb{R}^n$ be a set of m balls. Given $\mathcal{B}_{c,\rho}$ and $x \in \mathbb{R}^n$, the furthest point in $\mathcal{B}_{c,\rho}$ from x is given by $x^* = c + \rho(c - x)/\|c - x\|$, which can be computed in $O(n)$ operations. Therefore, each iteration of Algorithm 3.1 still requires $O(mn)$ operations, which implies that Algorithm 3.1 computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn/\epsilon)$ operations and returns an ϵ -core set of size $O(1/\epsilon)$. In addition to computing the furthest point at each iteration, Algorithm 4.1 also requires the computation of the closest point in a finite set. The size of this set is bounded above by $O(1/\epsilon)$, which implies that each iteration can be performed in $O(mn + n/\epsilon)$ operations. Therefore, Algorithm 4.1 can compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn/\epsilon + n/\epsilon^2)$ operations and returns an ϵ -core set of size $O(1/\epsilon)$.

5.2 Set of Ellipsoids

Let $\mathcal{A} = \{\mathcal{E}_1, \dots, \mathcal{E}_m\} \subset \mathbb{R}^n$ be a set of m ellipsoids given by $\mathcal{E}_i := \{x \in \mathbb{R}^n : (x - c^i)^T Q^i (x - c^i) \leq 1\}$, where $c^i \in \mathbb{R}^n$ and $Q^i \in \mathbb{R}^{n \times n}$ is symmetric and positive definite for $i = 1, \dots, m$. The furthest point in an ellipsoid from a given point can be computed using a tight semidefinite programming relaxation with a fixed number of constraints in $O(n^{O(1)})$ operations in the real number model of computation [37], where $O(1)$ denotes a universal constant greater than three. Therefore, Algorithm 3.1 computes a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn^{O(1)}/\epsilon)$ operations and returns an ϵ -core set of size $O(1/\epsilon)$. Similarly, Algorithm 4.1 can compute a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$ in $O(mn^{O(1)}/\epsilon + n^{O(1)}/\epsilon^2)$ operations and returns an ϵ -core set of size $O(1/\epsilon)$.

5.3 Set of Half Ellipsoids

\mathcal{H} is said to be a half ellipsoid if it is given by the intersection of an ellipsoid with a halfspace. Let $\mathcal{A} = \{\mathcal{H}_1, \dots, \mathcal{H}_m\}$, where $\mathcal{H}_i := \{x \in \mathbb{R}^n : (x - c^i)^T Q^i (x - c^i) \leq 1, (f^i)^T x \leq \omega^i\}$, where $c^i \in \mathbb{R}^n, f^i \in \mathbb{R}^n, \omega^i \in \mathbb{R}$, and $Q^i \in \mathbb{R}^{n \times n}$ is symmetric and positive definite for $i = 1, \dots, m$.

Sturm and Zhang [31] established that the maximization of any quadratic function over a half ellipsoid can be cast as a semidefinite programming problem with a fixed number of constraints similarly to quadratic optimization over an ellipsoid. Therefore, the asymptotic overall complexity bounds of Algorithms 3.1 and 4.1 are identical to those for the case of a set of ellipsoids. In particular, both algorithms return an ϵ -core set of size $O(1/\epsilon)$.

5.4 Set of Intersections of a Pair of Similar Ellipsoids

Two n -dimensional ellipsoids \mathcal{E}_1 and \mathcal{E}_2 are said to be similar if they both admit a representation using the same semidefinite matrix. This implies that the length and the alignment of the corresponding axes are the same. Let $\mathcal{A} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$, where $\mathcal{T}_i := \{x \in \mathbb{R}^n : (x - c^i)^T Q^i (x - c^i) \leq 1, (x - h^i)^T Q^i (x - h^i) \leq 1\}$, where $c^i \in \mathbb{R}^n, h^i \in \mathbb{R}^n, \omega^i \in \mathbb{R}$, and $Q^i \in \mathbb{R}^{n \times n}$ is symmetric and positive definite for $i = 1, \dots, m$. It follows from the results of [31] that any quadratic optimization problem over the intersection of a pair of similar ellipsoids can be decomposed into two quadratic optimization problems over two half ellipsoids. Therefore, the asymptotic complexity bounds of Algorithms 3.1 and 4.1 are identical to those for the case of a set of half ellipsoids. Similarly, both algorithms return an ϵ -core set of size $O(1/\epsilon)$.

5.5 Further Extensions

We have described several classes of more general input sets for which an approximate minimum enclosing ball can be computed in polynomial time (for fixed ϵ) using the appropriate extensions of Algorithms 3.1 and 4.1. Obviously, the results can be extended to input sets that are composed of a combination of elements from each of the above classes. In particular, it is remarkable that the existence of an ϵ -core set of size $O(1/\epsilon)$ extends to much more general classes of input sets including those with uncountably many elements.

Similarly to the discussion in [37], the extent of input sets to which similar algorithmic frameworks can be applied largely depends on the efficiency of the furthest point computation required at each iteration of each of the two algorithms. It is well-known that quadratic optimization over certain sets (such as polytopes defined by inequalities) is computationally intractable. Therefore, our algorithmic framework does not yield a polynomial-time algorithm for a set of polytopes. In summary, the discovery of polynomial-time routines for quadratic optimization over other classes of input sets may lead to further efficient generalizations of our algorithms.

6 Computational Results

In this section, we report the results of our computational experiments. We implemented Algorithms 3.1 and 4.1 in MATLAB. For the purposes of comparison, we also implemented the first-order algorithm of Bădoiu and Clarkson [2] (henceforth the BC Algorithm). Their simple algorithm starts by setting any arbitrary point $a^i \in \mathcal{A}$ as the initial center c^1 . At

n	m	Time			Core Set Size			Iterations		
		A1	A2	BC	A1	A2	BC	A1	A2	BC
10	500	0.06	0.03	0.12	4.2	3.9	5.2	168.7	44.5	435.5
10	1000	0.15	0.03	0.14	4.6	3.8	5.4	330.7	41.6	344.4
20	5000	1.7	0.36	3.11	5.9	5.2	7	246.8	46	464.2
20	10000	4.46	0.58	4.65	4.9	4.1	5.8	319.2	36.3	334.4
30	30000	27	6.45	24.59	8.6	6.8	9.1	446.4	103.6	409
50	50000	71.62	16.87	68.78	10.5	9.5	11.8	429.8	98.4	415.1
100	100000	287.99	77.74	268.11	15.9	14.5	16.6	451.7	119	422.6

Table 1: Computational Results ($\epsilon = 10^{-3}$)

iteration k , let a^{j_k} denote the furthest point from c^k , $k = 1, 2, \dots$. The center is updated according to the following relation:

$$c^{k+1} = [1 - 1/(k+1)]c^k + [1/(k+1)]a^{j_k}, \quad k = 1, 2, \dots$$

Bădiou and Clarkson establish that $1/\epsilon^2$ such updates suffice in order to obtain a $(1 + \epsilon)$ -approximation to $\text{MEB}(\mathcal{A})$. Note that each iteration requires $O(mn)$ operations, which yields an overall complexity bound of $O(mn/\epsilon^2)$.

Similarly to Algorithms 3.1 and 4.1, it is easy to verify that the BC Algorithm also generates a sequence of feasible solutions for the dual formulation of the minimum enclosing ball problem. Therefore, in order to have a fair and meaningful comparison, we employed the same termination criterion that we used for Algorithms 3.1 and 4.1 rather than running the BC update for $1/\epsilon^2$ times.

In contrast with Algorithms 3.1 and 4.1, the objective functions evaluated at the iterates generated by the BC algorithm are not monotonically increasing in general. Therefore, the analysis of the BC algorithm uses entirely different tools [2].

The computational experiments were carried out on a Pentium IV processor with a clock speed of 2.80 GHz and 512 MB RAM running under Linux. We used MATLAB version 7.3.0.298 (R2006b) in our experiments.

We used two data sets in our experiments. The first data set was randomly generated as in [1] with sizes (n, m) varying from $(10, 500)$ to $(100, 100000)$. In all of our experiments, we set $\epsilon = 10^{-3}$. For each fixed (n, m) , ten different data sets were generated and the results are reported in terms of the averages over these data sets in Table 1, which is divided into four sets of columns. The first set of columns reports the size (n, m) . The next three sets of columns present the CPU time, core set size, and the number of iterations, respectively. Each one of these three sets is further divided into three columns labeled A1, A2, and BC corresponding to Algorithm 3.1, Algorithm 4.1, and the BC algorithm, respectively.

As illustrated by Table 1, each of the three algorithms is capable of quickly computing an approximation to the minimum enclosing ball of the given input set. In particular, all three algorithms terminated under eight minutes even on the largest instances. In terms of CPU

time, Algorithm 4.1 has a significantly better performance than Algorithm 3.1 and the BC algorithm, both of which have similar running times. All three algorithms computed very small core sets of similar sizes. Algorithm 4.1 always returned the smallest core sets for each input set. The core sets computed by Algorithm 3.1 and the BC algorithm have similar sizes with the former being slightly better than the latter. In terms of the number of iterations, Algorithm 4.1 once again significantly outperforms the other two algorithms, both of which exhibit a similar performance.

A close examination of the computational results reveals that Algorithm 4.1 resulted in reductions of 73% to 88% in terms of running time and of 74% to 90% in terms of number of iterations in comparison with the other two algorithms. This improved behavior of Algorithm 4.1 resulting from the incorporation of away steps into Algorithm 3.1 was analyzed by Ahipařaođlu, Todd, and Sun [1], who establish the linear convergence of this algorithmic framework under fairly general assumptions. Furthermore, due to allowing points to be dropped from the working core set, the sizes of the core sets computed by Algorithm 4.1 are about 10% to 30% smaller than those returned by the other two algorithms.

The next data set we considered is the vertices of the unit simplex. Bădoiu and Clarkson [3] establish a tight upper bound of $\lceil 1/\epsilon \rceil$ on the size of the core set for such an input set under the assumption that $n \geq \lceil 1/\epsilon \rceil$. In an attempt to assess the performances of the three algorithms on such a data set, we considered the vertices of the unit simplex with $n = 1000$ using $\epsilon \in \{1, .1, .01, .001\}$. The results of this experiment are presented in Table 2, which is organized similarly to Table 1.

ϵ	Time			Core Set Size			Iterations		
	A1	A2	BC	A1	A2	BC	A1	A2	BC
1	.01	0	.01	2	2	2	0	0	1
.1	.03	.03	.02	11	11	11	9	9	10
.01	1.85	1.86	1.84	101	101	101	99	99	100
.001	183.06	181.97	182.48	1000	1000	1000	998	998	999

Table 2: Vertices of the unit simplex ($n = 1000$)

As illustrated by Table 2, all three algorithms have similar performances on the vertices of the unit simplex in \mathbb{R}^n with $n = 1000$. Note that both the size of the core set and the number of iterations grow proportionally to $1/\epsilon$. These results are in agreement with the tight core set bound of [3]. This example illustrates that the asymptotic bounds on the core set size and the number of iterations for Algorithms 3.1 and 4.1 in general cannot be improved. However, all three algorithms computed the exact minimum enclosing ball for $\epsilon = 10^{-3}$ (and for any $\epsilon \geq 10^{-3}$). Therefore, the upper bound of $\lceil 1/\epsilon \rceil$ on the size of the core set is no longer tight for $n \leq \lceil 1/\epsilon \rceil$.

7 Concluding Remarks

In this paper, we proposed and analyzed two algorithms that compute an approximation to the minimum enclosing ball of a given finite set of vectors. Both algorithms exploit the underlying geometric structure of the problem. Each of the two algorithms is especially well-suited for large-scale instances of the minimum enclosing ball problem for which a moderate approximation suffices. Both algorithms can compute a small core set whose size depends only on the approximation parameter. We have discussed how our algorithms can be extended to more general input sets without sacrificing the iteration complexity and hence the size of the core set. In particular, several classes of input sets admit small and finite core sets. Our computational experiments reveal that both of our algorithms are capable of quickly computing a good approximation to the minimum enclosing ball of a finite set of vectors. Algorithm 4.1, which is obtained by incorporating away steps into Algorithm 3.1, seems to exhibit a significantly better performance than other first-order algorithms. The sizes of the core sets computed by our algorithms are usually fairly small. The example that consists of the vertices of the unit simplex illustrates that our analysis in general cannot be improved.

While the discovery of efficient algorithms such as interior-point methods revolutionized convex optimization, the computational cost of each iteration of such algorithms quickly becomes prohibitive as the size of the problems increases. Therefore, it seems desirable to design specialized algorithms for large-scale problems that exploit the underlying special structure of the problem. We have developed two such algorithms for the minimum enclosing ball problem in this paper. We intend to continue our work on developing specialized algorithms for other classes of large-scale structured optimization problems in the near future.

References

- [1] D. Ahipasaoglu, P. Sun, and M. J. Todd. Linear convergence of a modified Frank-Wolfe algorithm for computing minimum-volume enclosing ellipsoids. Technical Report TR1452, Cornell University, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York, 2006.
- [2] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proceedings of the 14th Annual Symposium on Discrete Algorithms*, pages 801–802, 2003.
- [3] M. Bădoiu and K. L. Clarkson. Optimal core-sets for balls. Unpublished manuscript, 2006.
- [4] 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.

- [5] L. M. Blumenthal and G. E. Wahlin. On the spherical surface of smallest radius enclosing a bounded subset of n -dimensional Euclidean space. *Bulletin of the American Mathematical Society*, 47:771–777, 1941.
- [6] R. K. Chakraborty and P. K. Chaudhuri. Note on geometrical solutions for some minimax location problems. *Transportation Science*, 15:164–166, 1981.
- [7] J. A. Chatelon, D. W. Hearn, and T. J. Lowe. A subgradient algorithm for certain minimax and minisum location problems. *Mathematical Programming*, 15:130–145, 1978.
- [8] G. Chrystal. On the problem to construct the minimum circle enclosing n given points in the plane. In *Proceedings of the Edinburgh Mathematical Society*, volume 3, pages 30–33, 1885.
- [9] D. J. Elzinga and D. W. Hearn. The minimum covering sphere problem. *Management Science*, 19(1):96–104, 1972.
- [10] K. Fischer and B. Gärtner. The smallest enclosing ball of balls: Combinatorial structure and algorithms. *International Journal of Computational Geometry and Applications*, 14(4–5):341–378, 2004.
- [11] K. Fischer, B. Gärtner, and M. Kutz. Fast smallest-enclosing-ball computation in high dimensions. In *Algorithms–ESA*, volume 2832 of *Lecture Notes in Computer Science*, pages 630–641. Springer Berlin/Heidelberg, 2003.
- [12] R. L. Francis. Some aspects of a minimax location problem. *Operations Research*, 15(6):1163–1169, 1967.
- [13] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.
- [14] B. Gärtner. Fast and robust smallest enclosing balls. In *Proceedings of the 7th Annual European Symposium on Algorithms (ESA)*, volume 1643 of *Lecture Notes in Computer Science*, pages 325–338, 1999.
- [15] B. Gärtner and S. Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. In *Proceedings of the 16th Annual Symposium on Computational Geometry*, pages 110–118, 2000.
- [16] A. Goel, P. Indyk, and K. R. Varadarajan. Reductions among high-dimensional proximity problems. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 769–778, 2001.
- [17] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, 1988.

- [18] C. H. Guo, M. Y. Lu, J. T. Sun, and Y. C. Lu. A new algorithm for computing the minimal enclosing sphere in feature space. In *Fuzzy Systems and Knowledge Discovery*, volume 3614 of *Lecture Notes in Computer Science*, pages 196–204. Springer Berlin/Heidelberg, 2005.
- [19] D. W. Hearn and J. Vijay. Efficient algorithms for the (weighted) minimum circle problem. *Operations Research*, 30(4):777–795, 1981.
- [20] S. K. Jacobsen. An algorithm for the minimax Weber problem. *European Journal of Operational Research*, 6:144–148, 1981.
- [21] P. Kumar, J. S. B. Mitchell, and E. A. Yildirim. Approximate minimum enclosing balls in high dimensions using core-sets. *The ACM Journal of Experimental Algorithmics*, 8(1), 2003.
- [22] C. L. Lawson. The smallest covering cone or sphere. *SIAM Review*, pages 415–416, 1965.
- [23] N. Megiddo. Linear time algorithms for linear programming in \mathbb{R}^3 and related problems. In *Proceedings of the 23rd Annual Symposium on the Foundations of Computer Science*, pages 151–162, 1982.
- [24] K. P. K. Nair and R. Chandrasekaran. Optimal location of a single service center of certain types. *Naval Research Logistics Quarterly*, 18:503–510, 1971.
- [25] F. Nielsen and R. Nock. Approximating smallest enclosing balls. In *Computational Science and Its Applications*, volume 3045 of *Lecture Notes in Computer Science*, pages 147–157. Springer Berlin/Heidelberg, 2004.
- [26] F. Nielsen and R. Nock. A fast deterministic smallest enclosing disk approximation algorithm. *Information Processing Letters*, 93(6):263–268, 2005.
- [27] B. J. Oommen. An efficient geometric solution to the minimum spanning circle problem. *Operations Research*, 35(1):80–86, 1987.
- [28] S. H. Pan and X. S. Li. An efficient algorithm for the smallest enclosing ball problem in high dimensions. *Applied Mathematics and Computation*, 172(1):49–61, 2006.
- [29] R. Panigrahy. Minimum enclosing polytope in high dimensions. Unpublished manuscript, 2006.
- [30] M. I. Shamos. *Computational Geometry*. PhD thesis, Yale University, New Haven, Connecticut, 1978.
- [31] J. F. Sturm and S. Z. Zhang. On cones of nonnegative quadratic functions. *Mathematics of Operations Research*, 28:246–267, 2003.

- [32] J. J. Sylvester. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*, 1(79), 1857.
- [33] J. J. Sylvester. On Poncelet's approximate linear valuation of Surd forms. *Philosophical Magazine*, xx:203–222, 1860. Fourth Series.
- [34] M. J. Todd and E. A. Yildirim. On Khachiyan's algorithm for the computation of minimum volume enclosing ellipsoids. *Discrete Applied Mathematics*, 2007. To appear.
- [35] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, volume 555 of *Lecture Notes in Computer Science*, pages 359–370. Springer-Verlag, 1991.
- [36] S. Xu, R. M. Freund, and J. Sun. Solution methodologies for the smallest enclosing circle problem. *Computational Optimization and Applications*, 25(1–3):283–292, 2003.
- [37] E. A. Yildirim. On the minimum volume covering ellipsoid of ellipsoids. *SIAM Journal on Optimization*, 17(3):621–641, 2006.
- [38] G. Zhou, K. C. Toh, and B. Sun. Efficient algorithms for the smallest enclosing ball problem. *Computational Optimization and Applications*, 30(2):147–160, 2005.