Approximating the Radii of Point Sets*

Kasturi Varadarajan† S. Venkatesh‡ Yinyu Ye§ Jiawei Zhang¶

April 17, 2006

Abstract

We consider the problem of computing the outer-radii of point sets. In this problem, we are given integers n, d, k where $k \leq d$, and a set P of n points in R^d. The goal is to compute the *outer k-radius* of P, denoted by $R_k(P)$, which is the minimum, over all $(d-k)$-dimensional flats F, of $\max_{p \in P} d(p, F)$, where $d(p, F)$ is the Euclidean distance between the point p and flat F. Computing the radii of point sets is a fundamental problem in computational convexity with many significant applications. The problem admits a polynomial time algorithm when the dimension d is constant [16]. Here we are interested in the general case when the dimension d is not fixed and can be as large as n, where the problem becomes NP-hard even for $k = 1$.

It is known that $R_k(P)$ can be approximated in polynomial time by a factor of $(1 + \varepsilon)$, for any $\varepsilon > 0$, when $d - k$ is a fixed constant [8, 23]. A

*Research by the first author is supported by NSF CAREER award CCR-0237431. Research by the second author is supported by an NSERC discovery grant. Research by the third and fourth authors is supported by NSF grant DMI-0231600.

A preliminary version of this paper appeared as : (i) K. R. Varadarajan, S. Venkatesh, and J. Zhang, Approximating the radii of point sets in high dimensions, *Proc. 43rd IEEE Symp. Found. Comput. Sci.*, 2002, and (ii) Y. Ye and J. Zhang, An improved algorithm for approximating the radii of point sets, *Proc. APPROX*, 2003.

†Department of Computer Science, The University of Iowa, Iowa City, IA 52242-1419, email: kvaradar@cs.uiowa.edu, www: http://www.cs.uiowa.edu/~kvaradar/

‡Department of Computer Science, University of Victoria, PO Box 3055, STN CSC, Victoria, BC, Canada V8W 3P6, email: venkat@cs.uvic.ca, www: http://www.cs.uvic.ca/~venkat

§Management Science and Engineering and, by courtesy, Electrical Engineering, Stanford University, Stanford, CA 94305, USA, email: yinyu-ye@stanford.edu www: http://www.stanford.edu/~yyye/

¶IOMS-Operations Management, Stern School of Business, New York University, 44 W. 4th Street, Suite 8-66, New York, NY 10012-1126, email: jzhang@stern.nyu.edu www: www.stern.nyu.edu/~jzhang

polynomial time algorithm that guarantees a factor of $O(\sqrt{\log n})$ approximation for $R_1(P)$, the width of the point set P, is implied by the results of Nemirovski et al. [29] and Nesterov [28].

In this paper, we show that $R_k(P)$ can be approximated by a ratio of $O(\sqrt{\log n})$ for any $1 \leq k \leq d$, thus matching the previously best known ratio for approximating the special case $R_1(P)$, the width of point set P. Our algorithm is based on semidefinite programming relaxation with a new mixed deterministic and randomized rounding procedure.

We also prove an inapproximability result that gives evidence that our approximation algorithm is doing well for a large range of k. We show that there exists a constant $\delta > 0$ such that the following holds for any $0 < \varepsilon < 1$: there is no polynomial time algorithm that approximates $R_k(P)$ within $(\log n)^\delta$ for all k such that $k \leq d - d^\varepsilon$ unless NP $\subseteq$ DTIME $[2^{(\log m)^{O(1)}}]$. Our inapproximability result for $R_k(P)$ extends a previously known hardness result of Brieden [15], and is proved by modifying Brieden's construction using basic ideas from PCP theory.

1 Introduction

Computing the outer k-radius of a point set is a fundamental problem in computational convexity with applications in global optimization, data mining, statistics and clustering, and has received considerable attention in the computational geometry literature [21, 22, 23]. In this problem, we are given integers n, d, k where $k \leq d$, and a set P of n points in R^d. A *flat* or *affine subspace* F in R^d is specified by a point $q \in R^d$ and a linear subspace H; it is defined as $F = \{q + h | h \in H\}$. The dimension of the flat F is defined to be the dimension of the linear subspace H. For any flat F, let $R(P, F) = \max_{p \in P} d(p, F)$ denote the *radius* of the flat F with respect to P, where $d(p, F)$ is the Euclidean distance between the point p and flat F. The goal is to compute the *outer k-radius* of P, denoted by $R_k(P)$, which is the minimum of $R(P, F)$ over all $(d-k)$-dimensional flats F. A $(d-k)$-flat is simply a flat of dimension $d-k$. Roughly speaking, the outer k-radius $R_k(P)$ measures how well the point set P can be approximated by an affine subspace of dimension $d-k$. A few special cases of $R_k(P)$ which have received particular attention includes: $R_1(P)$, half of the *width* of P; $R_d(P)$, the radius of the minimum enclosing ball of P; and $R_{d-1}(P)$, the radius of the minimum enclosing cylinder of P.

When the dimension d is a fixed constant, $R_k(P)$ can be computed exactly in polynomial time [16]. It is also known that $R_k(P)$ can be approximated by a factor of $(1+\varepsilon)$, for any $\varepsilon > 0$, in $O(n + f_d(\frac{1}{\varepsilon}))$ time [7, 2], where f_d is a polynomial for every fixed d. In this paper, we are interested in the general scenario when the dimensions k and d are not fixed and d can be as large as n.

When the dimensions k and d are part of the input, the complexity of computing/approximating $R_k(P)$ depends on the parameter $d-k$. It is well-known that the problem is polynomial time solvable when $d-k=0$, i.e., the minimum enclosing ball of a set of points can be computed in polynomial time (Gritzmann and Klee [21]). Megiddo [27] shows that the problem of determining whether there is a line that intersects a set of balls is NP-hard. In his reduction, the balls have the same radius, which implies that computing the radius $R_{d-1}(P)$ of the min-enclosing cylinder of a set of points P is NP-hard. Bădoiu et al. [8] show that $R_{d-1}(P)$ can be approximated in polynomial time by a factor of $(1+\varepsilon)$, for any $\varepsilon > 0$. Har-Peled and Varadarajan [23, 24] generalize the result and show that $R_k(P)$ can be approximated by a factor of $(1+\varepsilon)$ for any $\varepsilon > 0$ when $d-k$ is constant [1].

More hardness results are known when $d-k$ becomes large or when k becomes small. Bodlaender et al. [11] show that the problem is NP-hard when $k=1$. This is true even for the case $n = d+1$ ([21]). Gritzmann and Klee [21] also show that it is NP-hard to compute $R_k(P)$ if $k \leq c \cdot d$, for any fixed $0 < c < 1$. These negative results are further improved by Brieden et al. [12] and Brieden [15], the latter of which has shown that it is NP-hard to approximate $R_1(P)$, the width of a point set, to within *any* constant factor.

On the positive side, the algorithms of Nemirovski et al. [28] and Nesterov [29] imply that $R_1(P)$, or equivalently the width of the point set P, can be approximated within a factor of $O(\sqrt{\log n})$. Another algorithm for approximating the width of a point set is given by Brieden et al. [13, 14] and their algorithm has a performance guarantee $\sqrt{d/\log d}$ that is measured in the dimension d. Their algorithm in fact works for any convex body given in terms of appropriate "oracles"; the number of calls to the oracle is polynomial in the dimension d. They also show that this is the best possible result in the oracle model even if randomization is allowed. (For the case of a set with n points, their algorithm actually gives a $\sqrt{d/\log n}$ approximation with poly(n) calls to the oracle.) It is not clear if their algorithm can be extended to computing $R_k(P)$.

The problem of efficiently computing low-rank approximation of matrices has received considerable attention recently; see [1, 6, 18] and the references cited in these papers. This problem corresponds to computing the best $(d-k)$-dimensional subspace that fits a point set, where the quality of a subspace is the *sum* of the square of the distance of each point from the flat. The problem is therefore related to the one we study in this paper, where the quality of a flat is the maximum over the point-flat distances. However, the low-rank approximation problem can be solved in polynomial time for any $1 \leq k \leq d$.

[1] Note that R_k^{opt} in [24] is the same as R_{d-k} in this paper

Our Results and an Overview

We show that $R_k(P)$ can be approximated in polynomial time by the factor of $O(\sqrt{\log n})$ for all $1 \leq k \leq d$, thereby generalizing the result of Nemirovski et al. [28] to all values of k. Our algorithm is based on semidefinite programming relaxation with a mixed deterministic and randomized rounding procedure, in contrast to all other purely randomized rounding procedures used for semidefinite programming approximation.

Generally speaking, the problem of computing $R_k(P)$ can be formulated as a quadratic minimization problem. Semidefinite programming (SDP) problems (where the unknowns are represented by positive semidefinite matrices) have recently been developed for approximating such problems; see, for example, Goemans and Williamson [19]. In the case of $k = 1$, computing $R_1(P)$ corresponds to a SDP problem plus an additional requirement that the rank of the unknown matrix equals 1. Removing the rank requirement, the SDP problem becomes a relaxation of the original problem and polynomially solvable for any given accuracy. Once obtaining an optimal solution, say X, of the SDP relaxation, one would like to generate a rank-1 matrix, say $\hat{X} = yy^T$, from X, where y is a column vector and serves as a solution to the original problem. Such rank reduction is called "rounding", and many rounding procedures are proposed and almost all of them are randomized, see, for example, [10].

One particular procedure has been proposed by Nemirovski et al. [28] which can be used for approximating $R_1(P)$. Their procedure is a simple randomized rounding that can be described as follows: an optimal solution X of the SDP relaxation, whose rank could be as large as d, can be represented as (e.g., by eigenvector decomposition)

$$X = \lambda_1 v_1 v_1^T + \lambda_2 v_2 v_2^T + \cdots + \lambda_d v_d v_d^T.$$

Then one can generate a single vector y by taking a random linear combination of the vectors $\sqrt{\lambda_1}v_1, \sqrt{\lambda_2}v_2, \cdots, \sqrt{\lambda_d}v_d$ where the coefficients of the combination takes values of -1 or 1 uniformly and independently.

For the case $k \geq 2$, the SDP relaxation that we describe is best viewed as a direct relaxation of the problem of computing $R_k(P)$, rather than one that is obtained via a quadratic program formulation of $R_k(P)$. We then need to generate k rank-1 matrices from X, the optimal solution of the SDP relaxation, such that

$$\hat{X} = \sum_{i=1}^{k} y_i y_i^T$$

where y_is are orthogonal to each other. Our rounding procedure works as follows.

Having obtained an optimal solution for the SDP relaxation with

$$X = \lambda_1 v_1 v_1^T + \lambda_2 v_2 v_2^T + \cdots + \lambda_d v_d v_d^T,$$

we deterministically partition the vectors $v_1, v_2, \cdots, v_d$ into k groups where group j may contain n_j vectors and each group can be seen as a single semidefinite matrix with rank n_j. We then generate one vector from each group using the randomized rounding procedure similar to that of Nemirovski et al. [28]. The k vectors generated by this rounding procedure will automatically satisfy the condition that any pair of them must be orthogonal to each other. We then manage to show that the quality of these vectors yields an approximation ratio of no more than $O(\sqrt{\log n})$.

We also prove an inapproximability result that gives evidence that our approximation algorithm is doing well for a large range of k. We show that there exists a constant $\delta > 0$ such that the following holds for any $0 < \varepsilon < 1$: there is no polynomial time algorithm that approximates $R_k(P)$ within $(\log n)^\delta$ for all k such that $k \leq d - d^\varepsilon$ unless NP $\subseteq \tilde{P}$. $\tilde{P}$ denotes the complexity class DTIME$[2^{(\log m)^{O(1)}}]$, which is sometimes referred to as deterministic quasi-polynomial time. That is, $\tilde{P}$ contains the set of all problems for which there is an algorithm that runs in time $2^{(\log m)^{O(1)}}$ on inputs of size m.

To prove the lower bound result, we start with a two-prover protocol for 3SAT in which the verifier has very low error probability. Such a protocol is obtained as a consequence of the PCP Theorem of Arora *et al.* [4, 5] and the parallel repetition theorem of Raz [30]. The construction of Brieden [15] then implies a reduction from Max-3SAT to Width Computation such that the ratio of the width of point sets that correspond to satisfiable instances to those that correspond to unsatisfiable instances is large. This separation gives us the inapproximability result for the width. This result can then be extended to an inapproximability result for $R_k(P)$ for a large range of k.

The remainder of the paper is organized as follows. In Section 2, we present our algorithm for approximating the outer k-radius $R_k(P)$ of a point set P. In Section 3, we describe our inapproximability results. We make some concluding remarks in Section 4.

2 Approximating the Radius

We now present the quadratic program formulation of the outer k-radius problem and its semidefinite programming relaxation. It will be helpful to first introduce some notations that will be used later. The trace of a given square matrix A, denoted by Tr(A), is the sum of the entries on the main diagonal of A. We use I to denote the identity matrix whose dimension will be clear in the context. The

inner product of two vectors p and q is denoted by $\langle p, q\rangle$. The 2-norm of a vector x, denoted by $\|x\|$, is defined by $\sqrt{\langle x, x\rangle}$. For a matrix X, we use the notation $X \succeq 0$ to mean that X is a positive semidefinite matrix. For simplicity, we assume that P is symmetric in the sense that if $p \in P$ then $-p \in P$. This is without loss of generality for the following reason: We may, by performing a translation if necessary, assume that $0 \in P$. Denote the set $\{-p | p \in P\}$ by $-P$ and let $Q = P \cup -P$. It is clear that $R_k(P) \leq R_k(Q) \leq 2R_k(P)$. Therefore, if we found a good approximation for $R_k(Q)$ then it must also be a good approximation for $R_k(P)$.

Since P is a symmetric point set, the best $(d-k)$-flat for P contains the origin so that it is a subspace. Thus, the square of $R_k(P)$ can be defined by the optimal value of the following quadratic minimization problem:

$$\begin{array}{rll} R_k(P)^2 := & \text{Minimize} & \alpha \\ & \text{Subject to} & \sum_{i=1}^{k} \langle p, x_i\rangle^2 \leq \alpha, \ \forall p \in P, \\ & & \|x_i\|^2 = 1, \ i = 1, ..., k, \\ & & \langle x_i, x_j\rangle = 0, \ \forall i \neq j. \end{array} \tag{1}$$

Assume that $x_1, x_2, \cdots, x_k \in R^d$ is the optimal solution of (1). Then one can easily verify that the matrix $X = x_1x_1^T + x_2x_2^T + \cdots + x_kx_k^T$ is a feasible solution for the following semidefinite program:

$$\begin{array}{rll} \alpha_k^* := & \text{Minimize} & \alpha \\ & \text{Subject to} & \text{Tr}(pp^TX) \ (= p^TXp) \leq \alpha, \ \forall p \in P, \\ & & \text{Tr}(X) = k, \\ & & I - X \succeq 0, \ X \succeq 0. \end{array} \tag{2}$$

It follows that $\alpha_k^* \leq R_k(P)^2$. The following lemma follows from the above observations.

Lemma 1 *There exists an integer $r \geq k$ such that we can compute, in polynomial time, r nonnegative reals $\lambda_1, \lambda_2, \cdots, \lambda_r$ and r orthogonal unit vectors $v_1, v_2, \cdots, v_r$ such that*

(i). $\sum_{i=1}^{r} \lambda_i = k$.

(ii). $\max_{1 \leq i \leq r} \lambda_i \leq 1$.

(iii). $\sum_{i=1}^{r} \lambda_i \langle p, v_i\rangle^2 \leq R_k(P)^2$, *for any* $p \in P$.

Proof: We solve the semidefinite program (2), and let X^* be an optimal solution of (2). We claim that the rank of X^*, say r, is at least k. This follows from the

fact that $\text{Tr}(X^*) = k$ and $I - X^* \succeq 0$. In other words, $\text{Tr}(X^*) = k$ implies that the sum of the eigenvalues of X^* is equal to k, and $I - X^* \succeq 0$ implies that the all eigenvalues are less than or equal to 1. Therefore, X^* has at least k non-zero eigenvalues, which implies that the rank of X^* is at least k. Let $\lambda_1, \lambda_2, \cdots, \lambda_r$ be the r nonnegative eigenvalues and $v_1, v_2, \cdots, v_r$ be the corresponding eigenvectors (see [28] page 466 on computing the eigenvalues and eigenvectors in polynomial time). Then we have $\sum_{i=1}^{r} \lambda_i = k$ and $\max_{1 \le i \le r} \lambda_i \le 1$. Furthermore, for any $p \in P$,

$$\sum_{i=1}^{r} \lambda_i \langle p, v_i \rangle^2 = \text{Tr}(pp^T \sum_{i=1}^{r} \lambda_i v_i v_i^T) = \text{Tr}(pp^T X^*) \le \alpha_k^* \le R_k(P)^2.$$

■

2.1 Deterministic First Rounding

In this section, we prove a lemma concerning how to deterministically group the eigenvalues and their eigenvectors. The proof of the lemma is elementary but it plays an important role for proving our main result.

Lemma 2 *The index set $\{1, 2, \cdots, r\}$ can be partitioned into k sets $I_1, I_2, \cdots, I_k$ such that for any $i : 1 \le i \le k$, $\sum_{j \in I_i} \lambda_j \ge \frac{1}{2}$.*

Proof: Recall that $\sum_{j=1}^{r} \lambda_j = k$ and $0 \le \lambda_j \le 1$ for all j. Without loss of generality, we can assume that $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_r$. Our partitioning algorithm is the same as the Longest-Processing-Time heuristic algorithm for parallel machine scheduling problem. The algorithm works as follows:

STEP 1. For $i = 1, 2, \cdots, k$, set $I_i = \emptyset$ and let $L_i = 0$. Let $I = \{1, 2, \cdots, r\}$.

STEP 2. While $I \neq \emptyset$

choose j from I with the smallest index;

choose set i with the smallest value L_i;

Let $I_i := I_i \cup \{j\}$, $L_i := L_i + \lambda_j$ and $I := I - \{j\}$.

It is clear that when the algorithm stops, the sets $I_1, I_2, \cdots, I_k$ are a partition of $\{1, 2, \cdots, r\}$. Now we prove the lemma by contradiction. Assume that there exists some t such that $\sum_{j \in I_t} \lambda_j < \frac{1}{2}$.

We now claim that, for all i, $\sum_{j \in I_i} \lambda_j \le 1$. Otherwise, suppose $\sum_{j \in I_{t'}} \lambda_j > 1$ for some t'. Note that $\lambda_j \le 1$ for every j and thus there are at least two eigenvalues are assigned to $I_{t'}$. Denote the last element within $I_{t'}$ by s'. It follows that $\sum_{j \in I_{t'}} \lambda_j - \lambda_{s'} = \sum_{j \in I_{t'} \setminus \{s'\}} \lambda_j \le \sum_{j \in I_t} \lambda_j$ since, otherwise, we would have not assigned $\lambda_{s'}$ to $I_{t'}$ in the algorithm. However, since $\sum_{j \in I_t} \lambda_j < \frac{1}{2}$, we must have $\sum_{j \in I_{t'}} \lambda_j - \lambda_{s'} = \sum_{j \in I_{t'} \setminus \{s'\}} \lambda_j < \frac{1}{2}$. Thus, $\lambda_{s'} > \sum_{j \in I_{t'}} \lambda_j - \frac{1}{2} > \frac{1}{2}$.

This is impossible since $\lambda_{s'}$ is the last eigenvalue assigned to $I_{t'}$, which implies $\lambda_{s'} \leq \lambda_j$ for every $j \in I_{t'}$, and we have already proved that there must exist an l such that $s' \neq l \in I_{t'}$ and $\lambda_l \leq \sum_{j \in I_{t'} \setminus \{s'\}} \lambda_j < \frac{1}{2}$. Therefore, $\sum_{j \in I_i} \lambda_j \leq 1$ for all i, and in particular $\sum_{j \in I_t} \lambda_j < \frac{1}{2}$. It follows that $\sum_{i=1}^{k} \sum_{j \in I_i} \lambda_j < k$. However, we know that since $I_1, I_2, \ldots, I_k$ is a partition of the index set $\{1, 2, \ldots, r\}$, $\sum_{i=1}^{k} \sum_{j \in I_i} \lambda_j = \sum_{j=1}^{r} \lambda_j = k$. This results in a contradiction. Therefore, such t does not exists and the proof is completed. ■

Notice that the running time of the partitioning algorithm is bounded by $O(r \cdot k)$.[2]

2.2 Randomized Second Rounding

Assume now that we have found $I_1, I_2, \cdots, I_k$. Then our next randomized rounding procedure works as follows.

STEP 1. Generate an r dimensional random vector ϕ such that each entry of ϕ takes value, independently, -1 or 1 with probability $\frac{1}{2}$ each way.

STEP 2. For $i = 1, 2, \cdots, k$, let

$$x_i = \frac{\sum_{j \in I_i} \phi_j \sqrt{\lambda_j} \cdot v_j}{\sqrt{\sum_{j \in I_i} \lambda_j}}.$$

The following Lemmas show that $x_1, x_2, \cdots, x_k$ form a feasible solution for the original problem. In other words, they are k orthogonal unit vectors.

Lemma 3 *For $i = 1, 2, \cdots, k$, $\|x_i\| = 1$.*

Proof: Recall that $\langle v_l, v_j \rangle = 0$ for any $l \neq j$ and $\|v_j\| = 1$. By definition,

$$\begin{aligned} \|x_i\|^2 &= \left\langle \frac{\sum_{j \in I_i} \phi_j \sqrt{\lambda_j} v_j}{\sqrt{\sum_{j \in I_i} \lambda_j}}, \frac{\sum_{j \in I_i} \phi_j \sqrt{\lambda_j} v_j}{\sqrt{\sum_{j \in I_i} \lambda_j}} \right\rangle \\ &= \frac{1}{\sum_{j \in I_i} \lambda_j} \sum_{j \in I_i} \langle \phi_j \sqrt{\lambda_j} v_j, \phi_j \sqrt{\lambda_j} v_j \rangle \\ &= \frac{1}{\sum_{j \in I_i} \lambda_j} \sum_{j \in I_i} (\phi_j)^2 \lambda_j \|v_j\|^2 \end{aligned}$$

[2] An alternative way of partitioning the eigenvalues is the following: First, put the eigenvalues that are greater than or equal to $1/2$ into distinct subsets. If the number of such eigenvalues, say l, is not less than k, then we are done. Otherwise, arbitrarily put the remaining eigenvalues into $k - l$ subsets such that the sum of eigenvalues in each subset is greater than or equal to $1/2$. This method was suggested by an anonymous referee of a preliminary version of this paper.

$$= \quad 1$$

■

Lemma 4 *If $s \neq t$ then $\langle x_s, x_t \rangle = 0$.*

Proof: Since for any $j \in I_s$ and $l \in I_t$, $\langle v_j, v_l \rangle = 0$,

$$\begin{aligned} & \langle x_s, x_t \rangle \\ = & \left\langle \frac{\sum_{j \in I_s} \phi_j \sqrt{\lambda_j} v_j}{\sqrt{\sum_{j \in I_s} \lambda_j}}, \frac{\sum_{j \in I_t} \phi_j \sqrt{\lambda_j} v_j}{\sqrt{\sum_{j \in I_t} \lambda_j}} \right\rangle \\ = & \frac{1}{\sqrt{\sum_{j \in I_s} \lambda_j \cdot \sum_{j \in I_t} \lambda_j}} \left\langle \sum_{j \in I_s} \phi_j \sqrt{\lambda_j} v_j, \sum_{j \in I_t} \phi_j \sqrt{\lambda_j} v_j \right\rangle \\ = & \ 0. \end{aligned}$$

■

Now we establish a bound on the performance of our algorithm. First, let us introduce Bernstein's Theorem (see, e.g., [28]), which is a form of the Chernoff Bound.

Lemma 5 *Let ϕ be a random vector whose entries are independent and either 1 or -1 with probability $\frac{1}{2}$ each way. Then, for any vector e and $\beta > 0$,*

$$\text{prob}\{\langle \phi, e \rangle^2 > \beta \|e\|^2\} < 2 \cdot \exp(-\frac{\beta}{2}).$$

Let $C_{ip} = \sum_{j \in I_i} \lambda_j \langle p, v_j \rangle^2$. Then we have

Lemma 6 *For each $i = 1, 2, \cdots, k$ and each $p \in P$, we have*

$$\text{prob}\{\langle p, x_i \rangle^2 > 12 \log(n) \cdot C_{ip}\} < \frac{2}{n^3}.$$

Proof: Given i and p, define a $|I_i|$ dimensional vector e such that its entries are $\sqrt{\lambda_j} \langle p, v_j \rangle$, $j \in I_i$, respectively. Furthermore, we define the $|I_i|$ dimensional vector $\phi|_{I_i}$ whose entries are those of ϕ with indices in I_i. First notice that

$$\|e\|^2 = \sum_{j \in I_i} (\sqrt{\lambda_j} \langle p, v_j \rangle)^2 = \sum_{j \in I_i} \lambda_j \cdot \langle p, v_j \rangle^2 = C_{ip}.$$

On the other hand, since $\sum_{j\in I_i}\lambda_j \geq \frac{1}{2}$,

$$\begin{aligned}
& \langle p, x_i\rangle^2 \\
= & \left\langle p, \frac{\sum_{j\in I_i}\sqrt{\lambda_j}v_j\phi_j}{\sqrt{\sum_{j\in I_i}\lambda_j}}\right\rangle^2 \\
\leq & 2\left\langle p, \sum_{j\in I_i}\sqrt{\lambda_j}v_j\phi_j\right\rangle^2 \\
= & 2\left(\sum_{j\in I_i}\sqrt{\lambda_j}\phi_j\langle p, v_j\rangle\right)^2 \\
= & 2\left\langle \phi|_{I_i}, e\right\rangle^2.
\end{aligned}$$

Thus

$$\text{prob}\{\langle p, x_i\rangle^2 > 12\log(n)C_{ip}\} \leq \text{prob}\{\langle \phi|_{I_i}, e\rangle^2 > 6\log(n)\|e\|^2\}.$$

Therefore, the conclusion of the lemma follows by using Lemma 5 and by letting $\beta = 6\log(n)$. ■

Theorem 1 *We can compute in polynomial time, a $(d-k)$-flat such that, with probability at least $1-\frac{2}{n}$, the distance between any point $p\in P$ and F is at most $\sqrt{12\log(n)}\cdot R_k(P)$.*

Proof: For given $i = 1, 2, \cdots, k$ and $p\in P$, consider the event

$$B_{ip} = \{\phi | \langle p, x_i\rangle^2 > 12\log(n)\cdot C_{ip}\}$$

and $B = \bigcup_{i,p} B_{ip}$. The probability that the event B happens is bounded by

$$\sum_{i,p}\text{prob}\{\langle p, x_i\rangle^2 > 12\log(n)\cdot C_{ip}\} < \frac{2kn}{n^3} \leq \frac{2}{n}.$$

If B does not happen, then for any i and p,

$$\langle p, x_i\rangle^2 \leq 12\log(n)\cdot C_{ip}.$$

Therefore, for each $p\in P$,

$$\sum_{i=1}^{k}\langle p, x_i\rangle^2 \leq 12\log(n)\sum_{i=1}^{k}C_{ip} \leq 12\log(n)\cdot R_k(P)^2.$$

The last inequality follows from Lemma 1. This completes the proof by taking F as the subspace which is orthogonal to the vectors $x_1, x_2, \cdots, x_k$. ■

3 The Inapproximability Results

We start with formal definitions of the problems that will be used in the sequence of reductions from 3-SAT to computing the outer k-radius $R_k(P)$ of a set P of points. Our starting point will be the classic 3-SAT problem, where we are given a 3-CNF formula and we want to know if there is an assignment to its variables that simultaneously satisfies all its clauses. The next problem we consider is the resricted quadratic programming problem as defined by Brieden [15].

Definition 3.1 (ζ-Restricted Quadratic Programming) *We are given non-negative integers $\lambda, \tau, \kappa, \sigma$ and non-negative rational numbers $c_{p,q,a,b}$ for $p \in [\lambda]$, $q \in [\tau]$, $a \in [\kappa]$ and $b \in [\sigma]$. (For a non-negative integer n, $[n]$ denotes the set $\{1, 2, \ldots, n\}$.) Our goal is to maximize*

$$f(x) = \sum_{p,q,a,b} c_{p,q,a,b} x_{p,a} y_{q,b}$$

over the polytope $P \subseteq \mathbb{R}^{\lambda\kappa+\tau\sigma}$ described by

$$\sum_{a \in [\kappa]} x_{p,a} = 1 \text{ for } p \in [\lambda],$$

$$\sum_{b \in [\sigma]} y_{q,b} = 1 \text{ for } q \in [\tau],$$

$$0 \leq x_{p,a} \leq 1 \text{ for } p \in [\lambda], a \in [\kappa],$$

$$0 \leq y_{q,b} \leq 1 \text{ for } q \in [\tau], b \in [\sigma].$$

We denote instances in which $\kappa, \sigma \leq \zeta$ and $\lambda, \tau \leq \Delta$ by ζ-restricted $QP[\Delta]$.

Definition 3.2 (Symmetric Full-Dimensional Norm Maximization) *We are given a string (n, m, A), where n and m are natural numbers, A is a rational $m \times n$ matrix. Our goal is to maximize*

$$f(x) = ||x||_2$$

over all vectors x that belong to the polytope $P = \{x | -1 \leq Ax \leq 1\}$. We denote an instance in which the number of rows of A is at most m and the number of columns is at most n by $NM[m, n]$.

We first prove an inapproximability result for the case $k = 1$ and later extend it to a large range of k using a simple reduction. The crux of the proof is the following lemma.

Lemma 3.1 *There is a constant $c > 1$ such that for any sufficiently large integer parameter $t \geq 1$, there is a reduction T from 3-SAT formulas of size m to computing R_1 for a point set of size $n = 2^{O(t2^{3t} \log m)}$ in $d = 2^{O(t \log m)}$ dimensions such that:*

1. *If ψ is satisfiable, then $R_1(T(\psi)) \geq w$ for some w.*
2. *If ψ is unsatisfiable, then $R_1(T(\psi)) \leq w'$ for some w'.*
3. *$\frac{w}{w'} \geq c^t$.*

This reduction, including the computation of w and w', runs in time $2^{O(t2^{3t} \log m)}$.

Proof:
The proof involves a sequence of three reductions.

From 3SAT to Quadratic Programming. Bellare and Rogaway [9, Section 4] give a reduction from 3-SAT to quadratic programming via a two-prover protocol for 3-SAT. We use their reduction, but in order to get the right parameters in the hardness of approximation result for quadratic programming, we need to replace the two prover protocol that they start off with by a different one. We now describe this two prover protocol for 3-SAT.

The two-prover protocol: Feige [17, Proposition 2.1.2] shows there exists a polynomial time reduction T from 3-CNF formulas to 3-CNF formulas such that each clause of $T(\psi)$ has exactly three literals (corresponding to three different variables) and each variable appears in exactly five clauses and furthermore

1. If ψ is satisfiable, then $T(\psi)$ is satisfiable.
2. If ψ is not satisfiable, then $T(\psi)$ is at most $(1 - \epsilon)$-satisfiable for some constant $0 < \epsilon < 1$. That is, any assignment satisfies at most a fraction $(1 - \varepsilon)$ of all clauses in $T(\psi)$.

Without the requirement that each variable appear in exactly five clauses, such a reduction is known to be a consequence of the PCP Theorem [4]. We now describe the steps taken by the verifier in the two-prover protocol.

Step 1: Convert ψ to $T(\psi)$.

Step 2: Choose t clauses uniformly at random (with replacement) from $T(\psi)$. Ask prover P_1 for an assignment to the variables in each clause chosen.

Step 3: From each chosen clause, choose one of the three variables in that clause uniformly at random. We get t *distinguished* variables, possibly with repititions. Ask the prover P_2 for an assignment to each of these t variables.

Step 4: Accept if for each chosen clause, it is satisfied by the assignment received from prover P_1, and the assignments made by the two provers to the distinguished variable from the clause are consistent. (Acceptance means that the verifier declares ψ to be satisfiable.)

By Raz's parallel repetition theorem [30], the error probabibility of this protocol, which is the probability that the verifier accepts a 3CNF formula ψ that is unsatisfiable, is bounded above by s^t for some $s < 1$. We refer the reader to Feige [17, Section 2.2] for a discussion of this and to the paper by Håstad [25] for more details on the use of two-prover protocols in inapproximability results. Also, note that in this protocol, the questions to the two provers are at most $O(t \log m)$ bits long where m is the input size, since $O(\log m)$ bits suffice to identify a clause or variable in $T(\psi)$. The answers from the two provers P_1 and P_2 are $3t$ and t bits long.

We now plug this two-prover protocol into the reduction of Bellare and Rogaway [9, Section 4] from SAT to restricted quadratic programming via two-prover protocols. Their description assumes for simplicity that the question and answer lengths of the two-prover protocol are the same, but their reduction works even if these sizes are different. Using the fact that the answer length is at most $3t$, we obtain[3]:

Lemma 3.2 (Bellare and Rogaway [9]) *There is a constant $f > 1$ such that for any sufficiently large integer $t \geq 1$, there is a reduction T_1 that maps 3-CNF formulas of size m to 2^{3t}-restricted $QP[2^{O(t \log m)}]$ such that:*

1. *If ψ is satisfiable, then $OPT(T_1(\psi)) = w_1$ for some w_1.*
2. *If ψ is unsatisfiable, then $OPT(T_2(\psi)) \leq w_2$ for some w_2.*
3. *$\frac{w_1}{w_2} \geq f^t$.*

Moreover, this reduction, including the computation of w_1 and w_2, runs in time $2^{O(t \log m)}$.

[3] In Bellare and Rogaway's reduction, the connection between the parameters of the two-prover protocol for 3-SAT and the parameters of the resulting ζ-restricted $QP[\Delta]$ instance is as follows: ζ is exponential in the answer length, Δ is exponential in the question length, and the "gap" f^t in Lemma 3.2 is the reciprocal of the error probability of the protocol.

From Quadratic Programming to Norm Maximization. Brieden ([15], Theorem 3.4) describes a sequence of interesting reductions that converts an instance of quadratic programming to an instance of the norm maximization problem. Using this reduction, we obtain:

Lemma 3.3 (Brieden [15]) *For any $\lambda > 0$, there is a reduction T_2 from restricted Quadratic Programming to Symmetric Full-dimensional Norm Maximization that maps 2^{3t}-restricted $QP[2^{O(t \log m)}]$ into $NM[2^{O(t2^{3t} \log m)}, 2^{O(t \log m)}]$ with the following property: for any input L of QP to T_2,*

$$\frac{OPT(L)}{(1+\lambda)} \leq OPT(T_2(L)) \leq (1+\lambda)OPT(L).$$

Moreover, the reduction T_2 runs in time $2^{O(t2^{3t} \log m)}$.

From Norm Maximization to Width Computation. The reduction from Symmetric Full-Dimensional Norm Maximization to Width computation is simple [20] and is in fact used by Brieden [15]. Let $a_i \in \mathbb{R}^n$ be the vector that corresponds to the i'th row of matrix A which is input to the norm-maximation problem, for $1 \leq i \leq m$. Thus the norm-maximization problem is

$$\begin{array}{rll} \gamma := & \text{Maximize} & ||x||_2 \\ & \text{Subject to} & \langle a_i, x \rangle^2 \leq 1, \text{ for } 1 \leq i \leq m. \end{array} \tag{3}$$

The reduction T_3 simply constructs a set B of points by adding, for each $1 \leq i \leq m$, the points a_i and $-a_i$ to B. Since B is a symmetric point set, $R_k(B)^2$ is given by the program

$$\begin{array}{ll} \text{Minimize} & \alpha \\ \text{Subject to} & \langle a_i, x \rangle^2 \leq \alpha, \text{ for } 1 \leq i \leq m, \\ & \|x\|^2 = 1 \end{array} \tag{4}$$

It is easy to verify that $\gamma = 1/R_k(B)$.

The reduction T claimed in Lemma 3.1 is obtained by composing the reductions T_1, T_2 and T_3. In particular, choose λ such that $(1+\lambda)^2 < f$, and let

$$c = \frac{1}{(1+\lambda)^2} f > 1.$$

It can now be checked that Lemma 3.1 holds with this choice of c. ■

Theorem 3.4 *1. There exists a constant $\delta > 0$ such that the following holds: there is no quasi-polynomial time algorithm that approximates $R_1(P)$ within $(\log n)^\delta$ unless NP $\subseteq \tilde{P}$.*

2. *Fix any constant $b \geq 1$. Then there is no quasi-polynomial time algorithm that approximates $R_1(P)$ within $(\log d)^b$ unless NP $\subseteq \tilde{P}$.*

Proof: To prove part 1, we apply the reduction of Lemma 3.1 with $t = \log\log m$ to obtain an instance of computing R_1 for a set of $n = 2^{O(t2^{3t}\log m)}$ in $d = 2^{O(t\log m)}$ dimensions. Choose $\delta' < \frac{\log c}{5}$. Then

$$\frac{c^t}{(t2^{3t})^{\delta'}} \geq \frac{c^t}{(2^{4t})^{\delta'}} \geq (\frac{c}{2^{4\delta'}})^t > (2^{\delta'})^t \geq (\log m)^{\delta'}.$$

Thus $c^t > (t2^{3t}\log m)^{\delta'}$. Since $n = 2^{O(t2^{3t}\log m)}$, we can choose $\delta < \delta'$ such that for n large enough,

$$c^t > (\log n)^{\delta}.$$

To prove part 2, we apply the reduction of Lemma 3.1 with $t = \frac{2p\log\log m}{\log c}$ for some sufficiently large constant p. Then,

$$\frac{c^t}{t^p} \geq \frac{2^{2p\log\log m}}{t^p} \geq 2^{p\log\log m}\frac{2^{p\log\log m}}{t^p} = 2^{p\log\log m}\left(\frac{\log m}{t}\right)^p > 2^{p\log\log m},$$

since $\log m > t$ for sufficiently large m.

Thus, $c^t > t^p 2^{p\log\log m} = (t\log m)^p$. Since the dimension d is $2^{O(t\log m)}$, it follows that for every constant $b \geq 1$, we can choose p large enough such that

$$c^t > (\log d)^b.$$

Observe that the reduction runs in quasi-polynomial time for our choice of t in both the cases and hence the theorem follows. ■

We now give the easy reduction from width to the outer k-radius that proves the main result of this section.

Theorem 3.5 *1. There exists a constant $\delta > 0$ such that the following holds for any $0 < \varepsilon < 1$: there is no quasi-polynomial time algorithm that approximates $R_k(P)$ within $(\log n)^\delta$ for all k such that $k \leq d - d^\varepsilon$ unless NP $\subseteq \tilde{P}$.*

2. *Fix any $\varepsilon > 0$. Fix any constant $c \geq 1$. Then there is no quasi-polynomial time algorithm that approximates $R_k(P)$ within $(\log d)^c$ for all k such that $k \leq d - d^\varepsilon$ unless NP $\subseteq$ DTIME $\tilde{P}$.*

Proof: Let P be a set of n points in $\mathbb{R}^d$. We map P to a set P' of n points in $\mathbb{R}^{d+k-1}$ using the function that takes a point $(x_1, \ldots, x_d) \in \mathbb{R}^d$ to the point $(x_1, \ldots, x_d, 0, \ldots, 0)$. It is easily checked that $R_1(P) = R_k(P')$. Theorem 3.5

follows from this reduction and some simple calculations: Observe that the reduction runs in polynomial time even if we set k to be $d^{1/\varepsilon} - d + 1$. With this choice, the target dimension $d' := d + k - 1$ equals $d^{1/\varepsilon}$. Thus $k = d^{1/\varepsilon} - d + 1 \geq d' - d'^{\varepsilon}$. Theorem 3.5 (1) now follows by applying Theorem 3.4 (1). For part (2), we apply Theorem 3.4 (2) with $b = 2c$. Since

$$(\log d)^{2c} \geq (\varepsilon \log d')^{2c} = (\log d'/eps^2)^c (\log d')^c \geq (\log d')^c$$

for sufficiently large d', Theorem 3.5 (2) also follows. ■

4 Conclusions

Finding efficient rounding methods for semidefinite programming relaxation plays a key role in constructing better approximation algorithms for various hard optimization problems. All of them developed to date are randomized in nature. Therefore, the mixed deterministic and randomized rounding procedure developed in this paper may have its own independent value. We expect to see more applications of the procedure in approximating various computational geometry and space embedding problems.

References

[1] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations In *Proc. ACM Symp. Theory of Computing*, 2001.

[2] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, Approximating extent measures of points, *J. ACM*, 51 (2004), 606–635.

[3] P.K. Agarwal and C.M. Procopiuc. Approximation algorithms for projective clustering. In *Proc. 11th ACM-SIAM Sympos. Discrete Algorithms*, pages 538–547, 2000.

[4] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Hardness of Approximation Problems. *Journal of ACM*, 45:3, 1998, 501–555.

[5] S. Arora and S. Safra. Probabilistic Checking of Proofs: a new characterization of NP. *Journal of ACM*, 45:1, 1998, 70–122.

[6] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral Analysis of Data. In *Proc. ACM Symp. Theory of Computing*, 2001.

[7] G. Barequet and S. Har-Peled, "Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions," *J. Algorithms*, 38:91-109, 2001.

[8] M. Bădoiu, S. Har-Peled and P. Indyk, "Approximate Clustering via Core-sets," In *Proc. ACM Symp. Theory of Computing*, 2002.

[9] M. Bellare and P. Rogaway. The complexity of approximating a nonlinear program. *Mathematical Programming B*, 69:3, 1995, 429–441.

[10] D. Bertsimas and Y. Ye, "Semidefinite relaxations, multivariate normal distributions, and order statistics," *Handbook of Combinatorial Optimization (Vol. 3)*, D.-Z. Du and P.M. Pardalos (Eds.) pp. 1-19, (1998 Kluwer Academic Publishers).

[11] H.L. Bodlaender, P. Gritzmann, V. Klee and J. Van Leeuwen, "The Computational Complexity of Norm Maximization", *Combinatorica*, 10: 203-225, 1990.

[12] A. Brieden, P. Gritzmann, and V. Klee, "Inapproximability of Some Geometric and Quadratic Optimization Problems," In P.M. Pardalos, editor, *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*, 96-115, Kluwer, 2000.

[13] A. Brieden, P. Gritzmann, R. Kannan, V. Klee, L. Lovasz and M. Simonovits, "Deterministic and Randomized Polynomial-time Approximation of Radii," To appear in *Mathematika*.

[14] A. Brieden, P. Gritzmann, R. Kannan, V. Klee, L. Lovasz and M. Simonovits, "Approximation of Diameters: Randomization Doesn't Help," In *Proc. IEEE Symp. Foundations of Comp. Sci.*, 244-251, 1998.

[15] A. Brieden, "Geometric Optimization Problems Likely Not Contained in APX," *Discrete Comput. Geom.,* 28:201-209, 2002.

[16] U. Faigle, W. Kern, and M. Streng, "Note on the Computational Complexity of j-Radii of Polytopes in R^n," *Mathematical Programming*, 73:1-5, 1996.

[17] U. Feige A Threshold of $\ln n$ for Approximating Set Cover. *J. ACM*, 45(4):634–652, 1998.

[18] A. Frieze, R. Kannan, and S. Vempala Fast Monte-Carlo algorithms for finding low rank approximations. *J. ACM*, 51(6):1025–1041, 2004.

[19] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semi-definite programming. *Journal of the ACM* 42, (1995), 1115–1145.

[20] P. Gritzmann and V. Klee, "Inner and Outer j-Radii of Convex Bodies in Finite-Dimensional Normed Spaces," *Discrete Comput. Geom.*, 7:255-280, 1992.

[21] P. Gritzmann and V. Klee, "Computational Complexity of Inner and Outer j-Radii of Polytopes in Finite-Dimensional Normed Spaces," *Math. Program.*, 59:162-213, 1993.

[22] P. Gritzmann and V. Klee, "On the Complexity of Some basic Problems in Computational Convexity: I. Containment Problems," *Discrete Math.*, 136:129-174, 1994.

[23] S. Har-Peled and K. Varadarajan, "Projective Clustering in High Dimensions Using Core-sets," In *Proc. ACM Symp. Comput. Geom.*, 2002.

[24] S. Har-Peled and K. Varadarajan, "High-Dimensional Shape Fitting in Linear Time," *Discrete & Computational Geometry*, 32(2): 269-288, 2004.

[25] J. Håstad. Some optimal inapproximability results. *Journal of ACM* 48: 798–859, 2001.

[26] W.B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz Mapping into Hilbert Space," *Contemporary Mathematics*, 26:189-206, 1984.

[27] N. Megiddo. On the complexity of some geometric problems in unbounded dimension. *Journal of Symbolic Computation*, 10:327–334, 1990.

[28] A. Nemirovski, C. Roos and T. Terlaky, "On Maximization of Quadratic Forms Over Intersection of Ellipsoids with Common Center," *Math. Program.*, 86:463-473, 1999.

[29] Yu. Nesterov, "Global Quadratic Optimization via Conic Relaxation," In eds H. Wolkowicz, R. Saigal and L. Vandenberghe, "Handbook of Semidefinite Programming Theory, Algorithms, and Applications", Kluwer Academic Publishers, Norwell, MA 02061 USA, 2000.

[30] R. Raz. A parallel repetition theorem. *SIAM Journal of Computing*, 27:3, 1998, 763–803.