On Approximating the Radii of Point Sets in High Dimensions

Kasturi R. Varadarajan* S. Venkatesh† Jiawei Zhang‡

April 27, 2002

Abstract

Let P be a set of n points in $\mathbb{R}^d$. For any $1 \leq k \leq d$, the *outer k-radius* of P, denoted by $R_k(P)$, 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. We consider the scenario when the dimension d is not fixed and can be as large as n. Computing the various radii of point sets is a fundamental problem in computational convexity with many applications (See [18]).

The main result of this paper is a randomized polynomial time algorithm that approximates $R_k(P)$ to within a factor of $O(\sqrt{\log n \cdot \log d})$ for any $1 \leq k \leq d$. This algorithm is obtained using techniques from semidefinite programming and dimension reduction. Previously, good approximation algorithms were known only for the case $k = 1$ and for the case when $k = d - c$ for any constant c; there are polynomial time algorithms that approximate $R_k(P)$ to within a factor of $(1+\varepsilon)$, for any $\varepsilon > 0$, when $d-k$ is any fixed constant [23, 7]. On the other hand, some results from the mathematical programming community on approximating certain kinds of quadratic programs [27, 26] imply an $O(\sqrt{\log n})$ approximation for $R_1(P)$, the width of the point set P.

We also prove an inapproximability result for computing $R_k(P)$, which easily yields the conclusion that our approximation algorithm performs quite well for a large range of values of k. Our inapproximability result for $R_k(P)$ improves the previous known hardness result of Brieden [13], and is proved by improving the parameters in Brieden's construction using basic ideas from PCP theory.

* Department of Computer Science, University of Iowa, kvaradar@cs.uiowa.edu
† DIMACS, Rutgers University, venkat@dimacs.rutgers.edu
‡ Department of Management Sciences, University of Iowa, jiazhang@dollar.biz.uiowa.edu

1 Introduction

We consider the problem of computing the various outer-radii of point sets in high dimensions. Let P be a set of n points in $\mathbb{R}^d$. For any $1 \le k \le d$, the *outer k-radius* of P, denoted by $R_k(P)$, 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. We note that $R_1(P)$ is the *width* of P, $R_d(P)$ is the radius of the minimum enclosing ball of P, and $R_{d-1}(P)$ is the radius of the minimum enclosing cylinder of P. Informally, the various radii measure how well the point set P can be approximated by lower dimensional affine subspaces. Computing the outer k-radius of a point set is a fundamental problem in computational convexity and has applications in data mining, statistics, and clustering scenarios [18, 19, 23].

The problem of computing the outer k-radius of a point set has received considerable attention in the computational geometry literature. The outer k-radius of a set P of n points can be computed exactly in polynomial time in fixed dimension [14]. It can also be approximated to within a factor of $(1+\varepsilon)$, for any $\varepsilon > 0$, in $O(n + (1/\varepsilon)^{O(dk)})$ time [6, 22]. Thus the problem is reasonably well understood when the dimension d is taken to be a fixed constant. These algorithms are not satisfactory when the dimension is large. In the rest of this section, we are interested in efficient algorithms when the dimension d can be as large as n.

When the dimension d is large, most of the previous results have focussed on the extreme cases when $k = 1$, $k = d$ and $k = d - 1$. Gritzmann and Klee [18] show that the minimum enclosing ball $(R_d(P))$ of a set of points can be computed in polynomial time. Badiou et al. [7] give a poly-time algorithm that computes a $(1+\varepsilon)$-approximation, for any $\varepsilon > 0$, of the minimum enclosing cylinder $(R_{d-1}(P))$ of a set of points. Har-Peled and Varadarajan [23] give a poly-time algorithm that computes a $(1+\varepsilon)$-approximation, for any $\varepsilon > 0$, to $R_k(P)$ whenever $d - k$ is a fixed constant. These results show that approximating $R_k(P)$ can be done efficiently for small $d - k$. To the best of our knowledge, it is not known whether computing $R_{d-1}(P)$ is NP-hard.

The problem seems to get harder when $d - k$ becomes becomes large. Bodlaender et al. [9] show that computing the width of a point set is NP-hard. Gritzmann and Klee [18] show that it is NP-hard to compute the width of even a set of $2d$ points. They also show that it is NP-hard to compute $R_k(P)$ for such small point sets as long as $k \le c \cdot d$, for any fixed $0 < c < 1$. Brieden et al. [10] show that it is NP-hard to approximate the width of a point set to within a factor of 12/11. Recently, Brieden [13] showed that is NP-hard to approximate the width of a point set to within *any* constant factor. Nesterov [27] gives an algorithm for approximating a certain kind of quadratic program that implies a poly-time algorithm for computing an $O(\sqrt{\log n})$ approximation of the width. Later, Nemirovski et al. [26] present a different algorithm for approximating a more general quadratic program that also imply the same result for the width. These two papers from the mathematical programming literature, which do not appear to be that well-known in the computational geometry literature, do not mention the consequences of their results for the width problem. Yet we think that the result that they imply for the width problem is quite remarkable.

Brieden et al. [11, 12] give a polynomial time algorithm that gives a $\sqrt{d/\log d}$ approximation for the width of a point set. 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 $R_k(P)$. However, by using the technique of approximating a polytope by its Lowner-John Ellipsoid [20], we can easily get a polynomial-time algorithm that gives an $O(\sqrt{d})$ approximation for $R_k(P)$. We remark that it is possible to combine the ideas of Brieden et al. [11, 12] and Nemirovski et al. [26] to give a polynomial time algorithm that gives a $d^{1/4}$ approximation for the width of the point set. For details, see Section 5.

The problem of efficiently computing low-rank approximation of matrices has received considerable attention recently; see [1, 5, 15] and the references cited in these papers. This problem corresponds to computing the best $(d-k)$-flat that fits a (symmetric) point set, where the quality of a flat 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.

Our Techniques and Results

Our main result is a polynomial time algorithm that approximates $R_k(P)$ within a factor of $O(\sqrt{\log n \cdot \log d})$ for any $1 \le k \le d$.

Theorem 1.1 *Given as input a point set $P \in \mathbb{R}^d$ and any integer $1 \le k \le d$, there is a randomized polynomial time algorithm that returns, with probability at least $1/2$, a $(d-k)$-flat F such that the distance of any point in P from F is at most $O(\sqrt{\log n \cdot \log d}) R_k(P)$.*

The starting point of our work is the paper of Nemirovski et al. [26], who present a polynomial time algorithm that gives an $O(\log n)$ approximation for the following quadratic program:

$$\begin{array}{ll} \text{Maximize} & x^T A x \\ \text{Subject to} & x^T A_i x \le 1, \text{ for } 1 \le i \le n \end{array}$$

where A is a symmetric $d \times d$ matrix, each A_i is a symmetric positive semidefinite $d \times d$ matrix, and $x \in \mathbb{R}^d$. For any set $P = \{p_1, \ldots, p_n\}$ of points in $\mathbb{R}^d$, the width $R_1(Q)$ of the symmetric point set $Q = P \cup -P$ is easily seen to be the minimum, over all unit vectors $u \in \mathbb{R}^d$, of $\max_{p_i \in P} |\langle p_i, u\rangle|$. Consider the special case of the quadratic program where A is set to be the identity matrix and A_i is set to be the matrix $p_i {p_i}^T$. It is easy to check that the value of the program is exactly $1/R_1(Q)^2$. Thus the algorithm of Nemirovski et al. yields an $O(\sqrt{\log n})$ approximation for the width of Q.

Nemirovski et al. solve the following semi-definite relaxation of the quadratic program:

$$\begin{array}{ll} \text{maximize} & \mathrm{Tr}(AX) \\ \text{Subject to} & \mathrm{Tr}(A_i X) \leq 1, \ \text{for } 1 \leq i \leq n \\ & \text{X is positive semi-definite} \end{array}$$

where X is constrained to be a symmetric positive semidefinite $d \times d$ matrix. They then use randomized rounding to turn the solution X of the SDP into a $O(\log n)$ approximation of the QP.

Our first attempt at the problem of computing $R_k(Q)$ was to write it as a quadratic program, solve the corresponding semi-definite relaxation, and then round the solution. But this approach does not work because of the presence of some difficult constraints in the quadratic program that say that some pairs of vectors must be orthogonal. We then made the simple but important observation that the semi-definite program, with $A = I$ and $A_i = p_i {p_i}^T$, is *directly* a relaxation of the problem of computing $R_k(Q)$ for any $1 \leq k \leq d$. But this was not good enough precisely because the semi-definite program is a relaxation for *every* k. In particular, because it is a relaxation for $k = 1$, it may give us nothing more than an approximation of the width $R_1(Q)$ of Q, and this is not what we want. We then observed that we can add a valid convex constraint to the semi-definite program that places an appropriate upper bound on the maximum eigenvalue of the matrix X. This simple constraint is crucial for the rounding step to work. We solve the resulting semi-definite program, round the solution X^* into a convenient form, and then apply the Johnson-Lindenstrauss dimension reduction technique [25] to reduce the rank of the solution X^* to $k + O(\log n * \log d)$. This gives us a low rank solution to the semi-definite program that is nearly as good as the original solution. From this low-rank solution, we obtain a $(d-k)$-flat that gives the desired approximation of $R_k(Q)$. It is interesting to note that unlike several recent papers (See the survey of Indyk [24]), we need the original version of the lemma proved by Johnson and Lindenstrauss. We need to orthogonally project vectors onto a subspace of the space they inhabit and we also need the lengths of the projections to be concentrated around the expected value.

We also prove an inapproximability result that gives evidence that our approximation algorithm is doing well for a large range of k.

Theorem 1.2 *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 $\subseteq$ DTIME $[2^{(\log m)^{O(1)}}]$.*

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 $[2^{(\log m)^{O(1)}}]$.

To prove the lower bound result for $R_k(P)$, we start with a two-prover protocol for Max-3SAT in which the verifier has very low soundness. Such a protocol is obtained as a consequence of the PCP Theorem of Arora *et al.* [3, 4] and the parallel repetition theorem of Raz [28]. The construction of Brieden [13] 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 width. This result can then be extended to an inapproximability result for $R_k(P)$ for a large range of k.

Organization of the paper

In Section 2, we show that, without loss of generality, we can move to a setting where the point set is symmetric and make some preliminary observations. In Section 3, we present our algorithm for approximating the outer k-radius of a symmetric point set. In Section 4, we present the lower bounds. We end with some remarks in Section 5.

2 Preliminaries

We will use $\mathrm{Tr}(A)$ to denote the trace of a symmetric matrix A. A flat of d-dimensional Euclidean space $\mathbb{R}^d$ is any affine subspace of $\mathbb{R}^d$. For any integer $0 \leq r \leq d$, an r-flat is any flat whose dimension is r. Let P be a set of n points in $\mathbb{R}^d$. We denote by $-P$ the set $\{-p \mid p \in P\}$. By performing a translation, if necessary, we may assume that that the convex hull $\mathrm{conv}(P)$ of P contains the origin. Let Q denote the point set $P \cup -P$. It is easy to see that $R_k(P) \leq R_k(Q) \leq 2R_k(P)$. We shall therefore try to find a good $(d-k)$-flat for Q.

Since Q is a symmetric point set, the best $(d-k)$-flat for Q contains the origin. For any point p and $(d-k)$-flat F passing through the origin, the distance of p from F is nothing but the length of the projection of p onto the subspace orthogonal to F. Thus we can define $R_k(Q)$ to be the minimum, over all sets $x_1, \ldots, x_k$ of k orthogonal unit vectors, of the square root of

$$\max_{p \in P} \sum_{i=1}^{k} \langle p, x_i \rangle^2.$$

It will be convenient to work with $R_k(Q)^2$ instead of $R_k(Q)$.

We will need the following consequence of the Johnson-Lindenstrauss lemma [25].

Lemma 2.1 *Let q be a vector in $\mathbb{R}^d$, let $v_1, \ldots, v_r$ be a set of orthogonal unit vectors. Let G be a random s-dimensional subspace of the subspace spanned by $v_1, \ldots, v_r$. Let $u_1, \ldots, u_s$ be an orthonormal basis for G. Then there exists a constant C such that if $s \geq C \log n$, the inequality*

$$\frac{r}{s} \cdot \frac{\langle q, u_1 \rangle^2 + \cdots + \langle q, u_s \rangle^2}{\langle q, v_1 \rangle^2 + \cdots + \langle q, v_r \rangle^2} \leq 2$$

holds with probability at least $1 - 1/n^2$.

This lemma has certain interesting consequences for the relation between the various radii of Q.

Proposition 2.2 *Let Q be a symmetric point set in $\mathbb{R}^d$ with n points. For any $1 \leq \ell \leq k \leq d$, $R_\ell(Q)^2 \leq \frac{2\max(\ell, C\log n)}{k} \cdot R_k(Q)^2$.*

In particular, if the width of Q is greater than a fraction $1/\sqrt{d}$ of the radius of the minimum enclosing ball of P ($R_1(Q) \geq R_d(Q)/\sqrt{d}$), then a random k-dimensional subspace gives, with high probability, an $O(\log n)$-approximation to $R_k(Q)^2$ for $k \geq C\log n$. It is also easy to see that a random k-dimensional subspace gives an $O(\log n)$-approximation for $k < C\log n$. Thus it is relatively straightforward to obtain the results of this paper when $\text{conv}(Q)$ is "fat". If $\text{conv}(Q)$ is not fat, for example when Q lies on a lower dimensional subspace, it is clear that a random k-dimensional subspace of $\mathbb{R}^d$ is not good enough.

3 The Algorithm

The algorithm involves three main steps: solving a semi-definite relaxation, rounding the solution obtained and then using dimension reduction to get a low-rank solution nearly as good as the original solution. We now describe each step in detail.

Semidefinite Relaxation

We use semidefinite programming to get a set of non-negative reals $\lambda_1, \ldots, \lambda_d$ and a set of orthogonal unit vectors $x_1, \ldots, x_d$ in $\mathbb{R}^d$ satisfying the constraints described in the following lemma.

Lemma 3.1 *Let $Q = P \cup -P$ be a set of $2n$ points in $\mathbb{R}^d$. We can compute, in polynomial time, a set of non-negative reals $\lambda_1, \ldots, \lambda_d$ and a set of orthogonal unit vectors $x_1, \ldots, x_d$ in $\mathbb{R}^d$ such that*

$$\sum_{i=1}^{d} \lambda_i = 1$$

$$\max_i \lambda_i \leq 1/k$$

$$\sum_{i=1}^{d} \lambda_i \langle p, x_i \rangle^2 \leq R_k(Q)^2/k \text{ for each } p \in P$$

Proof: We note first of all that the above is feasible, since by the definition of $R_k(P)$ there exist k orthogonal vectors $v_1, \ldots, v_k$ such that

$$\sum_{i=1}^{k} \langle p, v_i \rangle^2 \leq R_k(Q)^2 \text{ for each } p \in P.$$

To construct $\lambda_1, \ldots, \lambda_d$ and $x_1, \ldots, x_d$, we first solve the semi-definite program:

$$\text{Minimize } \alpha$$

$$\mathrm{Tr}(X) = 1$$

$$\mathrm{Tr}(pp^T X) \leq \alpha/k \text{ for each } p \in P$$

$$X \text{ is positive semi-definite}$$

$$\frac{1}{k} I - X \text{ is positive semi-definite}$$

X is required to be a $d \times d$ symmetric matrix. The last constraint (where I denotes the identity matrix) is equivalent to saying that the largest eigen-value of X should be at most $1/k$. The value of the optimal solution α^* to the semi-definite program is at most $R_k(Q)^2$, as is seen by letting $X = \frac{1}{k}(v_1 {v_1}^T + \cdots + v_k {v_k}^T)$.

Let X^* be the symmetric positive-definite matrix returned by the semi-definite program. We decompose it as $X^* = \lambda_1 x_1 {x_1}^T + \cdots \lambda_d x_d {x_d}^T$ where the x_i are orthogonal unit vectors and the λ_i are the eigenvalues of X^*. It is easy to verify that the x_i and the λ_i give us what we want. $\square$

Remark Clearly, we cannot hope to find the decomposition of X^* exactly in finite time. However, the decomposition of a "perturbation" of X^* can be found in polynomial time and this would be sufficient for our purposes. For more details, we refer the reader to [26], page 4. For simplicity, we assume that what we find is a decomposition of X^* itself.

Rounding the Solution

We now round the solution obtained in the previous step. This is described in the following lemma.

Lemma 3.2 *Let $Q = P \cup -P$ be a set of $2n$ points in $\mathbb{R}^d$. We can compute, in polynomial time, non-negative reals $\beta_1, \ldots, \beta_d$ and orthogonal unit vectors $x_1, \ldots, x_d$ such that each $\beta_i \in \{\frac{1}{m}, \frac{1}{2m}, \ldots, \frac{1}{2^{3 \log d} m}, 0\}$ for some $m \geq k$ and*

$$\sum_{i=1}^{d} \beta_i = 1$$

$$\sum_{i=1}^{d} \beta_i \langle p, x_i \rangle^2 \leq 4 R_k(Q)^2/k \textit{ for each } p \in P$$

Proof: Let $\lambda_1, \ldots, \lambda_d$ be the non-negative reals and $x_1, \ldots, x_d$ be the unit vectors returned by the algorithm of Lemma 3.1. (Essentially, we want to round each λ_i such that the rounded values belong to a small set of numbers.) If there are k or more λ_i's in the range $[1/2k, 1/k]$, we set $\lambda_i' = 1/2k$ for each such λ_i and $\lambda_i' = 0$ for every other λ_i. It is clear that

$$\sum_{i=1}^{d} \lambda_i' \geq 1/2$$

$$\max_i \lambda_i' \leq 1/2k$$

$$\sum_{i=1}^{d} \lambda_i' \langle p, x_i \rangle^2 \leq R_k(Q)^2/k \text{ for each } p \in P$$

By letting $\beta_i = \lambda_i' / (\sum_{i=1}^{d} \lambda_i')$, we obtain the desired result.

If there are at most $k-1$ λ_i's in the range $[1/2k, 1/k]$, we set $\lambda_i' = 1/2k$ for each such λ_i, $\lambda_i' = 0$ for every $\lambda_i \leq 1/d^3k$, and $\lambda_i' = \lambda_i$ for every other λ_i. It is easy to check that

$$\sum_{i=1}^{d} \lambda_i' \geq \frac{k-1}{2k} + \frac{1}{k} - \frac{d}{d^3 k} \geq 1/2$$

$$\max_i \lambda_i' \leq 1/2k$$

$$\sum_{i=1}^{d} \lambda_i' \langle p, x_i \rangle^2 \leq R_k(Q)^2/k \text{ for each } p \in P$$

Next, for each $1 \leq j \leq 3 \log d - 1$, we round each λ_i' in the range $(\frac{1}{2^{j+1}k}, \frac{1}{2^j k}]$ to $\frac{1}{2^{j+1}k}$. Since each non-zero λ_i' is smaller by a factor of at most $1/2$, the new λ_i' satisfy

$$\sum_{i=1}^{d} \lambda_i' \geq 1/4$$

$$\max_i \lambda_i' \leq 1/4k$$

$$\sum_{i=1}^{d} \lambda_i' \langle p, x_i \rangle^2 \leq R_k(Q)^2/k \text{ for each } p \in P$$

By letting $\beta_i = \lambda_i' / (\sum_{i=1}^{d} \lambda_i')$, we obtain the desired result. $\square$

Dimension Reduction

We now use dimension reduction to get a low-rank solution nearly as good as the original solution. This is given by the following lemma.

Lemma 3.3 *Let $Q = P \cup -P$ be a set of $2n$ points in $\mathbb{R}^d$. We can compute, in randomized polynomial time, a set $\gamma_1, \ldots, \gamma_\ell$ of non-negative real numbers and a set $u_1, \ldots, u_\ell$ of orthogonal unit vectors in $\mathbb{R}^d$ such that $\ell \leq k + O(\log n \log d)$ and with probability at least 1/2,*

$$\sum_{i=1}^{\ell} \gamma_i = 1$$

$$\max_i \gamma_i \leq 1/k$$

$$\sum_{i=1}^{\ell} \gamma_i \langle p, u_i \rangle^2 \leq 8 R_k(Q)^2/k \textit{ for each } p \in P.$$

Proof: We compute, in polynomial time, non-negative reals $\beta_1, \ldots, \beta_d$ and orthogonal unit vectors $x_1, \ldots, x_d$ in $\mathbb{R}^d$ satisfying the conditions of Lemma 3.2. It will be convenient to denote, for each $0 \leq j \leq 3 \log d$, the x_i's (if there are any) with corresponding $\beta_i = 1/(2^j m)$ by $v_j^1, \ldots, v_j^{n_j}$. (That is, n_j is the number of such x_i's.) We thus have

$$\sum_{j=0}^{3 \log d} \frac{n_j}{2^j m} = 1$$

$$\sum_{j=0}^{3 \log d} \frac{1}{2^j m} (\langle p, v_j^1 \rangle^2 + \cdots + \langle p, v_j^{n_j} \rangle^2) \leq 4 R_k(Q)^2/k \text{ for each } p \in P.$$

For each $0 \leq j \leq 3 \log d$, let d_j be the smallest integer such that

$$\frac{n_j}{m 2^j d_j} \leq 1/k.$$

Clearly, we have that

$$d_j \leq \frac{n_j k}{m 2^j} + 1.$$

Let $b_j = \min(n_j, \max(d_j, C \log n))$. Observe that

$$\frac{n_j}{m 2^j b_j} \leq 1/k.$$

For each $0 \leq j \leq 3\log d$, let $u_j^1, \dots, u_j^{b_j}$ be an orthonormal basis for a random b_j-dimensional subspace of $v_j^1, \dots, v_j^{n_j}$. (If $b_j = n_j$, this is just the original subspace spanned by $v_j^1, \dots, v_j^{n_j}$.) From Lemma 2.1, it follows that the inequality

$$\sum_{j=0}^{3\log d} \frac{n_j}{2^j m b_j}(\langle p, u_j^1\rangle^2 + \cdots + \langle p, u_j^{b_j}\rangle^2) \leq 8R_k(Q)^2/k$$

holds for every $p \in P$ with probability at least $1/2$. Furthermore, we have

$$\sum_{j=0}^{3\log d} \frac{n_j}{2^j m b_j} \cdot b_j = 1.$$

We have already checked that

$$\frac{n_j}{m 2^j b_j} \leq 1/k.$$

Finally, we have

$$\begin{aligned}
\sum_{j=0}^{3\log d} b_j &\leq \sum_{j=0}^{3\log d} \max\{C\log n, d_j\} \\
&\leq O(\log d \log n) + \sum_{j=0}^{3\log d} d_j \\
&\leq O(\log d \log n) + \sum_{j=0}^{3\log d} (\frac{n_j k}{m 2^j} + 1) \\
&\leq O(\log d \log n) + 3\log d + k \cdot \sum_{j=0}^{3\log d} \frac{n_j}{m 2^j} \\
&= O(\log d \log n) + 3\log d + k * 1 \\
&= O(\log d \log n) + k,
\end{aligned}$$

which completes the proof. □

The Main Result

We now get a good $(d-k)$-flat using the solution obtained after the dimension reduction step.

Lemma 3.4 *Let $Q = P \cup -P$ be a set of $2n$ points in $\mathbb{R}^d$. We can compute, in randomized polynomial time, a $(d-k)$-flat F such that with probability at least $1/2$ the distance of any point in Q from F is at most $O(\sqrt{\log n \cdot \log d})R_k(Q)$.*

Proof: Using the algorithm of Lemma 3.3, we compute a set of non-negative reals $\gamma_1, \ldots, \gamma_\ell$ and a set of orthonormal unit vectors $u_1, \ldots, u_\ell$ in $\mathbb{R}^d$ such that $\ell = k + O(\log d \log n)$ and

$$\sum_{i=1}^{\ell} \gamma_i = 1$$

$$\max_i \gamma_i \le 1/k$$

$$\sum_{i=1}^{\ell} \gamma_i \langle p, u_i \rangle^2 \le 8 R_k(Q)^2/k \text{ for each } p \in P.$$

Without loss of generality, we assume that $\gamma_1 \ge \gamma_2 \cdots \ge \gamma_\ell$. Now

$$\gamma_k \ge \frac{1 - (k-1)/k}{\ell - (k-1)} \ge \frac{1}{ck \log d \log n}$$

for some constant $c > 0$. We thus have

$$\sum_{i=1}^{k} \gamma_i \langle p, u_i \rangle^2 \le 8 R_k(Q)^2/k \text{ for each } p \in P,$$

and furthermore $\gamma_i \ge \frac{1}{ck \log d \log n}$ for $1 \le i \le k$. It follows from this that

$$\sum_{i=1}^{k} \langle p, u_i \rangle^2 \le (8c \log n \log d) R_k(Q)^2 \text{ for each } p \in P.$$

We have thus found the required $u_1, \ldots, u_k$. The $(d-k)$-flat we desire is the one orthogonal to these vectors. This also completes the proof of Theorem 1.1. $\square$

4 The Inapproximability Result

We start with formal definitions of problems that will be used in the sequence of reductions from Max-3SAT to computing the width $R_1(P)$ of a set of points.

Definition 4.1 [Quadratic Programming] We are given non-negative integers P, Q, A, B, non-negative rational numbers $c_{p,q,a,b}$ for $p \in [P]$, $q \in [Q]$, $a \in [A]$ and $b \in [B]$. 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 $P \subseteq \Re^{PA+QB}$ described by

$$\sum_a x_{p,a} = 1 \text{ for } p \in [P],$$
$$\sum_b y_{q,b} = 1 \text{ for } q \in [Q],$$
$$0 \leq x_{p,a} \leq 1 \text{ for } p \in [P], a \in [A],$$
$$0 \leq y_{q,b} \leq 1 \text{ for } q \in [Q], b \in [B].$$

We denote instances in which the number of inequalities is at most m and the number of variables is at most n by $QP[m, n]$.

Definition 4.2 [Norm Maximization] We are given a string (n, m, A, b), where n and m are natural numbers, A is a rational $m \times n$ matrix and a rational m-vector b. Our goal is to maximize

$$f(x) = ||x||_2^2$$

over all vectors x that belong to the polytope P given by $Ax \leq b$. We denote an instance in which the number of inequalities is at most m and the number of variables is at most n by $NM[m, n]$.

From now on, let $\tilde{P}$ denote the complexity class, quasi-polynomial time. That is, the set of all problems for which there are algorithms that run in time $2^{(\log m)^{O(1)}}$ on inputs of size m. We first prove Theorem 1.2 for the case $k = 1$ and later extend it to the range of k mentioned in the statement.

Theorem 4.3 *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 $NP \subseteq DTIME\ [2^{(\log m)^{O(1)}}]$.*

2. Fix any constant $c \geq 1$. Then there is no quasi-polynomial time algorithm that approximates $R_1(P)$ within $(\log d)^c$ unless $NP \subseteq DTIME\ [2^{(\log m)^{O(1)}}]$.

Proof: The proof involves showing a quasi-polynomial time reduction from the problem of MAX-3SAT to computing R_1 on a point set. The inapproximability result will then follow from the NP-completeness of Max-3SAT. The details of this reduction is given by the following lemma:

Lemma 4.4 *There is a reduction T from 3-SAT formulas of size m to solving R_1 on a point set of size $n = 2^{O(t2^t \log m)}$ 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 $R(T(\psi)) \leq w'$ for some w'.*
3. *$\frac{w}{w'} \geq c^t$ for some fixed constant $c > 1$.*

This reduction runs in time $2^{O(t2^t \log m)}$.

Proof of Theorem: We first prove Theorem 1.2 assuming Lemma 4.4 and give the proof of Lemma 4.4 later. To prove part 1 of Theorem 1.2, choose $t = \log\log m$ and choose $\delta < \frac{\log c}{3}$. Then

$$\begin{aligned} \frac{c^t}{(t2^t)^\delta} &\geq (\frac{c}{2^{2\delta}})^t \\ &> (2^\delta)^t \\ &\geq (\log m)^\delta. \end{aligned}$$

Hence,

$$c^t \geq (\log n)^\delta.$$

To prove part 2 of Theorem 1.2, choose $t > \frac{2b\log\log m}{\log c}$. Then,

$$\begin{aligned} \frac{c^t}{t^b} &> 2^{b\log\log m} \\ &= (\log m)^b. \end{aligned}$$

Hence,

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

Observe that the reduction runs in quasi-polynomial time for our choice of t in both the cases and hence Theorem 1.2 follows.

We are now going to prove Lemma 4.4 by a sequence of three reductions.

From Max-3SAT to Quadratic Programming Reduction from Max-3SAT to Quadratic Programming involves building a two-prover protocol for 3SAT formulas with low soundness.

The two-prover protocol: By the PCP Theorem [3], there exists polynomial time reduction T from 3SAT formulas to 3SAT formulas such that each clause of $T(\psi)$ has exactly three literals and each variables appears in 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.

We now describe the main steps of the two-prover protocol.

Step 1: Convert ψ to $T(\psi)$.

Step 2: Choose t random clauses from $T(\psi)$ and t variables at random, one from each of the t clauses.

Step 3: Ask prover P_1 for the assignment to each clause chosen. Ask the prover P_2 for the assignment to the each variable chosen.

Step 4: Accept if all the clauses are satisfied and the two assignments are consistent.

By Raz's parallel repetition theorem [28], the soundness of this protocol is bounded above by f^t for some $f < 1$. Also, note that in this protocol, the questions to the two provers are at most $t \log m$ bits long where m is the input size and the answers from the two provers P_1 and P_2 are $3t$ and t bits long. For more details on use of two-prover protocols in inapproximability results, see the paper by Håstad [21].

Lemma 4.5 (Bellare and Rogoway [8]) *Using the two-prover protocol described above, one can describe a reduction T_1 from 3SAT formulas of size m to $QP[2^{O(t \log m)}, 2^{O(t \log m)}]$ 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. *$\frac{w_1}{w_2} \geq f^t$ for some fixed constant $f > 1$.*

Moreover, this reduction runs in time $2^{O(t \log m)}$.

From Quadratic Programming to Norm Maximization We use the construction of Brieden. Brieden describes a sequence of interesting reductions that converts an instance of quadratic programming to an instance of the norm maximization problem.

Lemma 4.6 (Brieden [13]) *There is a reduction T_2 from Quadratic Programming to the Norm Maximization that maps $QP[2^{O(t \log m)}, 2^{O(t \log m)}]$ into $NM[2^{O(t 2^t \log m)}, 2^{O(t \log m)}]$ with the following property: for any input L of QP to T_2 and for any $\epsilon > 0$,*

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

Moreover, the reduction T_2 runs in time $2^{O(t 2^t \log m)}$ and the inequalities in $T_2(L)$ describe a centrally-symmetric polytope, that is, a polytope P such that $P = -P$.

From Norm Maximization to Width Computation The reduction from Norm Maximization to Width computation is simple [17]. The point set $P = \{p_1, \ldots, p_n\}$ where each point p_i is obtained from an inequality of the form $< p_i, x > \leq 1$. Note that the number of points in the point set P is the same as the number of inequalities in the Norm Maximization problem and the dimension is also preserved. In addition, it can be checked that the value of the norm maximization problem is $1/R_1(P)^2$. Hence, maximizing the norm of a point inside the centrally symmetric polytope reduces to computing the width R_1 of the symmetric point set P. The reduction described above runs in time $2^{O(t 2^t \log m)}$.

The reduction T required in Lemma 4.4 is obtained by composing the reductions T_1, T_2 and T_3. In particular, choose

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

It can be checked that Lemma 4.4 holds with this choice of c. This completes the proof of the theorem 4.3. □

We now give the easy reduction from width to the outer k-radius that proves Theorem 1.2.

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 1.2 follows from this reduction and some simple calculations. □

5 Remarks

The paper of Brieden et al. [12] implies a polynomial time algorithm that gives a $\sqrt{d/\log n}$ approximation for the width of an n-point set. We observe that this result can be improved to

Theorem 5.1 *There is a polynomial time algorithm that gives a $d^{1/4}$ approximation for the width of a point set.*

This result is obtained by noting that if the number of points n is at least $e^{\sqrt{d}}$, then the result of Nemirovski et al. [26] gives a $d^{1/4}$ approximation algorithm. If the number of points n is less than $e^{\sqrt{d}}$, then Theorem 3.3 of [12] implies a $d^{1/4}$ approximation algorithm.

Acknowledgments We would like to thank Andreas Brieden for sending us a preprint of his paper.

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

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

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

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

[6] 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.

[7] M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. To appear in *STOC 2002*.

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

[9] H. L. Bodlaender, P. Gritzmann, V. Klee, and J. van Leeuwen. The computational complexity of norm-maximization *Combinatorica* 10 (1990) 203–225.

[10] 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*, pages 96–115, Kluwer, 2000.

[11] 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.*

[12] 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.*, pages 244–251, 1998.

[13] A. Brieden. On geometric optimization problems likely not contained in APX. 2002, To appear.

[14] U. Faigle, W. Kern, and M. Streng. Note on the computational complexity of j-radii of polytopes in $\mathbb{R}^n$. *Mathematical Programming* 73:1–5, 1996.

[15] A. Frieze, R. Kannan, and S. Vempala Fast Monte-Carlo algorithms for finding low rank approximations. In *Proc. IEEE Symp. Foundations of Comput. Sci.*, 370–378, 1998.

[16] 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.

[17] 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.

[18] P. Gritzmann and V. Klee. Computational complexity of inner and outer j-radii of polytopes in finite-dimensional normed spaces. *Math. Program.*, 59:163–213, 1993.

[19] P. Gritzmann and V. Klee. On the complexity of some basic problems in computational convexity: I. containment problems. *Discrete Math.*, 136:129–174, 1994.

[20] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics.* Springer-Verlag, Berlin Heidelberg, 2nd edition, 1988. 2nd edition 1994.

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

[22] S. Har-Peled and K.R. Varadarajan. Approximate shape fitting via linearization. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 66–73, 2001.

[23] S. Har-Peled and K. Varadarajan. Projective clustering in high dimensions using core-sets. To appear in *ACM Symp. Comput. Geom.*, 2002.

[24] P. Indyk. Algorithmic applications of low-distortion geometric embeddings. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 10–31, 2001.

[25] W.B. Johnson and J. Lindenstrauss. Extensions of lipshitz mapping into hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[26] A. Nemirovski and C. Roos and T. Terlaky. On maximization of quadratic forms over intersection of ellipsoids with common center. *Mathematical Programming* Ser. A, 86:463–473, 1999.

[27] Yu. Nesterov. Global quadratic optimization via conic relaxation. Working paper, CORE, Catholic University of Louvaine, Belgium, 1998.

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