Equivalence of Convex Problem Geometry and Computational Complexity in the Separation Oracle Model

by

Robert M. Freund
Jorge Veraz

OR 383-09 **January 2009**

Equivalence of Convex Problem Geometry and Computational Complexity in the Separation Oracle Model[*]

Robert M. Freund[†]and Jorge Vera[‡]

January 2009

Abstract

Consider the following supposedly-simple problem:

$$\text{compute } x \text{ satisfying } x \in S \ ,$$

where S is a convex set conveyed by a separation oracle, with no further information (e.g., no bounding ball containing or intersecting S, etc.). Our interest in this problem stems from fundamental issues involving the interplay of (i) the *computational complexity* of computing a point $x \in S$, (ii) the *geometry* of S, and (iii) the *stability* or *conditioning* of S under perturbation. Under suitable definitions of these terms, we show herein that problem instances with favorable geometry have favorable computational complexity, validating conventional wisdom. We also show a converse of this implication, by showing that there exist problem instances in certain families characterized by unfavorable geometry, that require more computational effort to solve. This in turn leads, under certain assumptions, to a form of equivalence among computational complexity, the geometry of S, and the conditioning of S. Our measures of the geometry of S, relative to a given (reference) point $\bar{x}$, are the *aspect ratio* $A = R/r$, as well as R and $1/r$, where $B(\bar{x}, R) \cap S$ contains a ball of radius r. The *aspect ratio* arises in the analyses of many algorithms for convex problems, and its importance in convex algorithm analysis has been well-known for several decades. However, the terms R and $1/r$ in our complexity results are a bit counter-intuitive; nevertheless, we show that the computational complexity must involve these terms in addition to the aspect ratio even when the aspect ratio itself is small. This lower-bound complexity analysis relies on simple features of the separation oracle model of conveying S; if we instead assume that S is conveyed by a self-concordant barrier function, then it is an open challenge to prove such complexity lower-bound.

[*]This research has been partially supported through the Singapore-MIT Alliance and AFOSR Grant FA9550-08-1-0350

[†]MIT Sloan School of Management, 50 Memorial Drive, Cambridge, MA 02142, USA, email: rfreund@mit.edu

[‡]Dept. de Ingeniería Industrial y Sistemas, Facultad de Ingeniería, Pontificia Universidad Católica de Chile, Campus San Joaquín, Vicuña Mackenna 4860 Santiago, CHILE, email: jvera@ing.puc.cl

1 Introduction, Motivation, and Discussion

Consider the following supposedly-simple problem:

$$\text{compute } x \text{ satisfying } x \in S \ , \tag{1}$$

where $S \subset X$ is a convex set (bounded or not, open or not) conveyed by a separation oracle with no further information (e.g., no bounding ball containing or intersecting S, etc.), and X is a (finite) n-dimensional vector space. Our interest in (1) stems from fundamental issues involving the interplay of three notions: (i) the *computational complexity* of computing a point $x \in S$, (ii) the *geometry* of S, and (iii) the *stability* or *conditioning* of S. In this paper we focus on the equivalence of computational complexity and a suitable measure of the geometry of S, which leads under certain assumptions to an equivalence of all three notions.

There are two standard information models for convex sets, the separation oracle model and the (self-concordant) barrier model. A separation oracle for S, see [12], is a subroutine that, given a point $\hat{x}$ as input, returns the statement "$\hat{x} \in S$" if indeed this is the case, or if $\hat{x} \notin S$, returns a hyperplane H with the property that $\hat{x} \in H^-$ and $S \subset H^{++}$. Here $H \subset \Re^n$ denotes a hyperplane, and H^- and H^+ (H^{--} and H^{++}) denote the two closed (open) halfspaces of $\Re^n$ bounded by H. From a computational viewpoint, the information about H^+ is described by a nonzero vector $h \in \Re^n$ and scalar α for which $H^+ = \{x \in \Re^n : h^T x \geq \alpha\}$. For any separation oracle based algorithm, we define the number of iterations of the algorithm to be the number of oracle calls, i.e., we use oracle calls and iterations interchangeably.

The separation oracle model applies readily to the case when S is the feasible region of a conic linear system:

$$S := \{x \in \Re^n : b - Ax \in K\} \ , \tag{2}$$

where A is a linear operator from X to a (finite) m-dimensional vector space Y, $b \in Y$, $K \subset Y$ is a closed convex cone, and $d = (A, b)$ is the data for the system. In this case a separation oracle for the cone K can be used to easily construct a separation oracle for S.

1.1 Discussion of Main Results

A representative separation oracle algorithm for solving (1) is the ellipsoid algorithm [12], see also [6]. The ellipsoid algorithm is easy to implement and has very good theoretical complexity. We assume the reader is familiar with the concepts underlying this algorithm. In consideration of this, suppose that we know *a priori* a vector $\bar{x}$ and a positive scalar R with the property that $B(\bar{x}, R) \cap S$ has positive volume (where $B(c, \rho)$ denotes the ball of radius ρ centered at c). This last assumption can also be stated as saying that there exist values R and $r > 0$ for which it holds that:

$$B(\bar{x}, R) \cap S \text{ contains a ball of radius } r \ . \tag{3}$$

Then the ellipsoid algorithm for solving (1) can be initiated with the ball $B(\bar{x}, R)$, and the number of oracle calls (equivalently, the number of iterations) of the algorithm required to solve problem (1) is known (see [12]) to be at most

$$\left\lceil 2n^2 \ln\left(\frac{R}{r}\right)\right\rceil \ .$$

We will in general refer to a ratio of the form $A := R/r$ where R, r satisfy (3) as an *aspect ratio* of S relative to the reference point $\bar{x}$. The aspect ratio arises in the analysis of many algorithms for convex problems, and its importance in convex algorithm analysis dates back at least four decades, see e.g., Rosenblatt's 1962 book [18].

The ellipsoid algorithm requires prior specification of $\bar{x}$ and R to determine the starting ball (ellipsoid) with which to start the method; the condition that (3) holds for some $r > 0$ is also needed in order to guarantee convergence. However, at least in theory as well as in the case of typical instances of (1) when S is in the conic format (2), *a priori* values of $\bar{x}$ and R satisfying (3) are either not known or involve excessively conservative (large) values of $R >> 0$ for special structure and data representation of S. For example, suppose $S = \{x \in \Re^n : Ax \leq b\}$ for known rational data $d = (A, b)$ whose bit-encoding is of length L. Then one can pre-determine values of $\bar{x} := 0$ $(\in \Re^n)$, $R := 2^{nL}$, and $r := 2^{-nL}$ for which $B(\bar{x}, R) \cap S$ contains a ball of radius r if and only if (1) has an interior solution. This observation underlies Khachiyan's proof that linear optimization is polynomial-time in the bit-model in [7] (see also [2] and [6]); however in most instances there are significantly better values of R and r than those above. As another example, suppose $S = \{x \in \Re^n : Q_1x_1 + \ldots + Q_nx_n \succeq Q_0\}$ where $Q_i \in S^m$ are integral symmetric matrices of order m, for $i = 0, \ldots, n$. Porkolab and Khachiyan [9] show that if S is bounded then $S \subset B(0, R)$ for $R = e^{(Lm^{O(\min\{n,m^2\})})}$, where L is the maximum bit-length of the coefficients of the matrices $Q_0, \ldots, Q_n$. While this bound on R is pre-determined, it is doubly exponential in at least one of the dimensions of the problem. In Section 2 we present an extension of the ellipsoid algorithm that eliminates the need for any *a priori* information about R and/or r, and we prove an upper bound on the number of iterations required to solve (1) using this extension of the ellipsoid algorithm, given by:

$$\left\lceil 2(n+1)\left(\frac{1}{2}\ln(n) + 1.16 + n\ln\left(\frac{1}{r} + \frac{R}{r}\right) + \ln(1+R)\right)\right\rceil \ ,$$

where R, r are *any* existential values satisfying (3) but are not required to be known to run the algorithm (see Theorem 2.2). Notice that this iteration bound involves not just the aspect ratio R/r but also involves both $1/r$ and R itself. The complexity bound essentially states that sets S with favorable geometry relative to $\bar{x}$, in the sense that none of the quantities R/r, R, or $1/r$ are very large, will not require too much computation to solve.

The presence of the aspect ratio term $A = R/r$ in the above complexity bound is to be expected, especially in light of its presence in the case when R is known and given to the algorithm *a priori*. This still begs the question of whether the terms R and $1/r$ must be present or whether they can be removed from the complexity upper bound using a different algorithm and/or a different analysis. Notice that even if R/r is small, say at most an absolute constant, the values of R and/or $1/r$ might still be very large, yielding perhaps an overly conservative iteration complexity upper bound, as shown in the following two examples.

Example 1.1 *Let $n = 1$, $\bar{x} = 0$, and $S = [10^{-6}, 3 \times 10^{-6}]$. Then $r = 10^{-6}$, $R = 3 \times 10^{-6}$ yields $R/r = 3$ but $1/r = 10^6$.*

Example 1.2 *Let $n = 1$, $\bar{x} = 0$, and $S = [10^6, 3 \times 10^6]$. Then $r = 10^6$, $R = 3 \times 10^6$ yields $R/r = 3$ but $R = 3 \times 10^6$.*

Queried a different way, are all three geometric measures R/r, R, and $1/r$ necessary components of the computational complexity of solving (1)? We resolve this question in Section 3, where we show that the dependence of the complexity bound on R and $1/r$ (as well as on R/r) cannot be removed. In Theorems 3.1, 3.2, and 3.3, we show under suitable assumptions for some specific families of problem instances, that any separation oracle algorithm for solving (1) must, for some instance in the requisite family, require at least $\lfloor \log_2(R/r) - 1 \rfloor$, or $\lfloor \log_2 \log_2(R+1) \rfloor$, or $\lfloor \log_2 \log_2(1/4r) \rfloor$ iterations, respectively. While these lower bounds are not of the same order as the upper bound presented above, they do involve the same three geometric quantities R/r, R, and $1/r$.

Taken together, these results demonstrate a certain equivalence between computational complexity and problem instance geometry of S as measured by R/r, R, and $1/r$. Indeed, while problems with favorable geometry do not require too much computational effort to solve, there exist problem instances in certain families with unfavorable geometry, that require more computational effort to solve. This equivalence ties in nicely with results regarding the interplay of stability and conditioning, problem geometry, and computational complexity for problems in conic format (2). Considering S defined by (2) for data $d = (A, b)$ and keeping the cone K fixed, we measure the condition number of (2) using the "distance to infeasibility" which we now briefly review. Let $L(X, Y)$ denote the space of linear operators from X to Y and let $\mathcal{M} \subset L(X, Y) \times Y$ denote those data pairs $d = (A, b)$ for which S given by (2) is nonempty. For $d = (A, b) \in \mathcal{M}$, let $\rho(d)$ denote the "distance to infeasibility" for (2), namely:

$$\rho(d) := \min_{\Delta d = (\Delta A, \Delta b)} \{\|\Delta d\| : d + \Delta d \notin \mathcal{M}\} \ ,$$

under suitably defined norms on spaces and operators, see [14]. Then $\rho(d)$ denotes the smallest perturbation of our given data d which would render the resulting system in (2) infeasible. Next let $\mathcal{C}(d)$ denote the *condition measure* of (2), namely $\mathcal{C}(d) = \|d\|/\rho(d)$, which is a scale-invariant reciprocal of the distance to infeasibility. There are strong connections between $\mathcal{C}(d)$ and bounds on the stability of S under data perturbation, see Renegar [14]. It is shown in Renegar [15] and in [3] that problems with favorable condition numbers $\mathcal{C}(d)$ do not require too much computational effort to solve using interior-point methods and the ellipsoid method, respectively. Also, using $\bar{x} = 0$, it is shown in [4] that a favorable value of $\mathcal{C}(d)$ implies favorable geometry of S, namely favorable values of R/r, R, and $1/r$, and that a converse of this statement holds under an assumption of "conic curvature" defined and shown in [1]. Taken together, these cited results in combination with the results developed in this paper demonstrate an equivalence between favorable geometry, favorable complexity, and favorable conditioning under suitable assumptions.

Last of all, we remark that we have only shown an equivalence between the geometry of S and computational complexity in the separation oracle model, and not in the (self-concordant) barrier model. It is shown in [5] that favorable geometry implies favorable computational complexity of an interior-point method. However, it is an open challenge to prove an assertion that unfavorable geometry implies (say, in the worst case) a large number of iterations of an interior-point method.

The rest of the paper is organized as follows. In Sections 2 and 3 we present our lower and upper bounds on the complexity of solving (1), respectively. Section 4 discusses possible extensions and/or strengthening of the complexity bounds.

Notation. We assume that X is equipped with an inner-product norm $\|v\| := \sqrt{\langle v, v\rangle}$. For convenience we identify X with $\Re^n$ and the inner product $\langle \cdot, \cdot \rangle$ with the standard scalar product $\langle v, w\rangle = v^T w = \sum_{j=1}^n v_j w_j$. Let S^k, S^k_+, and S^k_{++} denote the set of symmetric matrices, symmetric positive semidefinite matrices, and symmetric positive definite matrices of order k, respectively. Let "$\succeq$" denote the Löwner partial ordering on symmetric matrices: $A \succeq B$ if and only if $A - B$ is positive semidefinite. Let Q^k denote the k-dimensional second-order cone $\{x \in \Re^k : \|(x_2, \ldots, x_k)\|_2 \le x_1\}$ for $k \ge 2$.

2 Upper Bounds on Complexity of (1) via an Extension of the Ellipsoid Algorithm

We first review some basic results about the ellipsoid algorithm, see Nemirovsky and Yudin [12], also Grötschel et al. [6], and then present an extension of the ellipsoid algorithm that solves (1) in the absence of any bounding information about S. The ellipsoid method applied to solve (1) for a given convex set $S \subset \Re^n$ is completely specified by the separation oracle for S and the center x^0 and shape matrix $Q_0 \succ 0$ of the starting ellipsoid $E^0 := \{x \in \Re^n : (x - x^0)^T Q_0 (x - x^0) \le 1\}$. By a simple change of variables, there is no loss of generality in assuming that $Q_0 = (\rho^0)^{-2} I$ for some

$\rho^0 > 0$, whereby $E^0 = B(x^0, \rho^0)$ and the information content of the starting ellipsoid is simply the center x^0 and radius ρ^0 of the starting ball.

Suppose that we know *a priori* a vector $\bar{x}$ and a positive scalar R with the property that $B(\bar{x}, R) \cap S$ has positive volume. Then the ellipsoid algorithm for solving (1) can be initiated with the ball $B(\bar{x}, R)$. The following is a generic result about the performance of the ellipsoid algorithm, where in the statement of the theorem "vol(T)" denotes the volume of a set T:

Theorem 2.1 Ellipsoid Algorithm Theorem with known R, from [12], also [6]. *Suppose that a vector $\bar{x}$ and a positive scalar R are known with the property that the set $B(\bar{x}, R) \cap S$ has positive volume. Then if the ellipsoid algorithm is initiated with the Euclidean ball $B(\bar{x}, R)$, the algorithm will compute a solution of (1) in at most*

$$\left\lceil 2n \ln\left(\frac{\mathrm{vol}(B(\bar{x}, R))}{\mathrm{vol}(B(\bar{x}, R) \cap S)}\right)\right\rceil$$

iterations, where each iteration makes one call of the separation oracle for S.

It is often convenient to restate this result using radii of certain balls rather than volumes of certain sets. The supposition of positive volume in this theorem is equivalent to the condition that there exist values R and $r > 0$ that satisfy (3). Then one then obtains, for example:

Corollary 2.1 (see [12]) *Suppose that S is given via a separation oracle, and that the ellipsoid algorithm is applied to solve (1) starting with $E^0 := B(\bar{x}, R)$ for some given $\bar{x}$ and R. If S, R, and r satisfy (3) for some $r > 0$, then the ellipsoid method will solve (1) in at most*

$$\left\lceil 2n^2 \ln\left(\frac{R}{r}\right)\right\rceil$$

iterations, where each iteration makes one call of the separation oracle.

Proof: Let $v(n)$ denote the volume of the n-dimensional unit ball, namely:

$$v(n) = \frac{\pi^{n/2}}{\Gamma(n/2 + 1)} \ , \tag{4}$$

see [6]. Letting $B(y, r)$ denote the ball of radius r contained in $B(\bar{x}, R) \cap S$, we have

$$\frac{\mathrm{vol}(B(\bar{x}, R))}{\mathrm{vol}(B(\bar{x}, R) \cap S)} \le \frac{\mathrm{vol}(B(\bar{x}, R))}{\mathrm{vol}(B(\bar{y}, r))} = \left(\frac{v(n)R^n}{v(n)r^n}\right) = (R/r)^n \ ,$$

and the result follows using Theorem 2.1.∎

Note that the ellipsoid algorithm requires prior specification of $\bar{x}$ and R just to implement the method; the condition that (3) holds for some $r > 0$ is also needed in order to guarantee convergence. Of course, *a priori* values of $\bar{x}$ and R for which (3) is true (for some $r > 0$) are typically either not known or involve excessively conservative (large) values of $R >> 0$ for special structure and data representation of S (see the discussion in Section 1.1). Except for instances such as these where prior information about special structure and data for S is given, such bounds on R are not known. Despite this lack of prior information in general, one can still utilize the ellipsoid algorithm for solving (1) by using a standard "lift and conify" transformation, see [3], that we now review and extend.

For any given value of $\bar{x}$ (think $\bar{x} = 0$ for convenience), define:

$$W^{\bar{x}} := \{(w, \theta) \in \Re^{n+1} : w - \bar{x} \in \theta(S - \bar{x}),\ \theta > 0\} \ . \tag{5}$$

Notice that $W^{\bar{x}} \cap \{(w, \theta) : \theta = 1\} = S \times \{1\}$, i.e., the restriction of $W^{\bar{x}}$ to the slice of $(x, \theta) \in \Re^{n+1}$ defined by $\theta = 1$ is simply S. Also, $W^{\bar{x}}$ is a (translated) convex cone in $\Re^{n+1}$ with base $(\bar{x}, 0)$. Therefore $W^{\bar{x}}$ is constructed by first lifting S to $S \times \{1\} \subset \Re^{n+1}$, and then conically extending $S \times \{1\}$ using the base point $(\bar{x}, 0)$, hence the term "lift and conify." One can solve (1) by instead working with the following equivalent problem in one higher dimension:

$$\text{compute } (w, \theta) \text{ satisfying } (w, \theta) \in W^{\bar{x}} \ . \tag{6}$$

The equivalence of (1) and (6) follows since solutions of one system can be converted to solutions of the other system as follows:

$$\begin{array}{rcl} (w, \theta) \in W^{\bar{x}} & \Rightarrow & x := \bar{x} + (w - \bar{x})/\theta \in S \\ x \in S & \Rightarrow & (w, \theta) := (\alpha(x - \bar{x}) + \bar{x}, \alpha) \in W^{\bar{x}} \text{ for all } \alpha > 0 \ . \end{array} \tag{7}$$

Furthermore, a separation oracle for S can be readily converted to a separation oracle for $W^{\bar{x}}$, as follows. If $(\hat{w}, \hat{\theta})$ is a given point, first check if $\hat{\theta} > 0$; if not, then $H^+ := \{(w, \theta) : \theta \geq 0\}$ is the requisite separating halfspace. If $\hat{\theta} > 0$, next check if $\hat{x} := \bar{x} + (\hat{w} - \bar{x})/\hat{\theta}$ is in S. If so, then $(\hat{w}, \hat{\theta}) \in W^{\bar{x}}$ and we are done. Otherwise, the separation oracle for S outputs $h \neq 0$ for which $h^T x \geq h^T \hat{x}$ for all $x \in S$, which then implies that $h^T\left(\bar{x} + (w - \bar{x})/\theta\right) \geq h^T\left(\bar{x} + (\hat{w} - \bar{x})/\hat{\theta}\right)$ for all $(w, \theta) \in W^{\bar{x}}$. Simplifying yields $H^+ := \{(w, \theta) : \hat{\theta} h^T w - \hat{\theta} h^T \bar{x} \geq \theta(h^T \hat{w} - h^T \bar{x})\}$ as the requisite separating halfspace for $(\hat{w}, \hat{\theta})$ in this case.

Because $W^{\bar{x}}$ is a (translated) convex cone in $\Re^{n+1}$ with base $(\bar{x}, 0)$, $W^{\bar{x}}$ contains points in the $(n + 1)$-dimensional unit ball centered at $(\bar{x}, 0)$, which we denote by $B^{\bar{x}}_{n+1}$:

$$B^{\bar{x}}_{n+1} := B_{n+1}((\bar{x}, 0), 1) := \left\{(w, \theta) \in \Re^{n+1} \mid \sqrt{(w - \bar{x})^T (w - \bar{x}) + \theta^2} \leq 1\right\} \ .$$

Therefore, given only $\bar{x}$, one can apply the ellipsoid algorithm to solve (6) (and hence solve (1)) using the initial ball $B^{\bar{x}}_{n+1}$, yielding the following "extended" version of the basic ellipsoid algorithm:

```
Extended Ellipsoid Method for Solving (1) with Unknown $R$
(a) Input: separation oracle for $S$, and initiating point $\bar{x} \in \Re^n$.
(b) If $\bar{x} \in S$, output $\bar{x}$ and Stop. Otherwise construct $W^{\bar{x}}$ using (5) and run the ellipsoid
        algorithm in $\Re^{n+1}$ starting with $B^{\bar{x}}_{n+1}$ to compute a point in $W^{\bar{x}}$. Output $(\hat{w}, \hat{\theta}) \in W^{\bar{x}}$.
(c) Set $\hat{x} \leftarrow \bar{x} + (\hat{w} - \bar{x})/\hat{\theta}$ and Stop.
```

In order to prove a complexity bound using the above extension of the ellipsoid method, we must bound the ratio of the volume of $B^{\bar{x}}_{n+1}$ to the volume of $B^{\bar{x}}_{n+1} \cap W^{\bar{x}}$. This is accomplished in the following lemma, a variant of which was first presented in [3]:

Lemma 2.1 *Suppose that $S \cap B(\bar{x}, R)$ contains a ball of radius $r > 0$. Then*

$$\ln\left(\frac{\mathrm{vol}\left(B^{\bar{x}}_{n+1}\right)}{\mathrm{vol}\left(B^{\bar{x}}_{n+1} \cap W^{\bar{x}}\right)}\right) \le \frac{1}{2}\ln(n) + 1.16 + n\ln\left(\frac{1}{r} + \frac{R}{r}\right) + \ln(1+R) .$$

Proof: We prove a slightly more general result which will imply the conclusion of Lemma 2.1 as a special case. Consider the slightly more general definition of $W^{\bar{x}}$ parameterized by a scalar $c > 0$:

$$W^{\bar{x}}_c := \{(w, \theta) \in \Re^{n+1} : c \cdot (w - \bar{x}) \in \theta(S - \bar{x}),\ \theta > 0\} , \tag{8}$$

and note that we recover $W^{\bar{x}}$ by setting $c = 1$. Notice that $W^{\bar{x}}_c \cap \{(w, \theta) : \theta = c\} = S \times \{c\}$, i.e., the restriction of $W^{\bar{x}}_c$ to the slice of $(x, \theta) \in \Re^{n+1}$ defined by $\theta = c$ is simply S. We will prove:

$$\ln\left(\frac{\mathrm{vol}\left(B^{\bar{x}}_{n+1}\right)}{\mathrm{vol}\left(B^{\bar{x}}_{n+1} \cap W^{\bar{x}}_c\right)}\right) \le \frac{1}{2}\ln(n) + 1.16 + n\ln\left(\frac{\sqrt{R^2+c^2}}{r}\right) + \ln\left(\frac{\sqrt{R^2+c^2}}{c}\right) . \tag{9}$$

Notice that Lemma 2.1 follows immediately from (9) by setting $c = 1$ and using the inequality $\sqrt{R^2+1} \le R + 1$.

We now prove (9). By hypothesis there exists y for which $B(y, r) \subset S \cap B(\bar{x}, R)$. By performing a translation if necessary, we can assume that $\bar{x} = 0$, which simplifies the arithmetic manipulations below. Define $H := W^{\bar{x}}_c \cap B^{\bar{x}}_{n+1} = \{(w, \theta) : c \cdot w/\theta \in S,\ \theta > 0,\ \|(w, \theta)\|_2 \le 1\}$ and $T := \{(w, \theta) : c \cdot w/\theta \in B(y, r),\ 0 < \theta \le c\}$. Defining $\delta := \sqrt{R^2 + c^2}$, we first prove that $T \subset B((0, 0), \delta)$. To see why this is true, let $(w,\theta) \in T$, then

$$\begin{aligned}
\|(w,\theta)\|_2 &= \|(w-(\theta/c)y+(\theta/c)y,\theta)\|_2 \\
&= \sqrt{\|(w-(\theta/c)y+(\theta/c)y\|_2^2+\theta^2} \\
&\leq \sqrt{(\|(w-(\theta/c)y\|_2+\|(\theta/c)y\|_2)^2+\theta^2} \\
&\leq \sqrt{((\theta/c)r+(\theta/c)\|y\|_2)^2+\theta^2} \\
&\leq (\theta/c)\sqrt{R^2+c^2} \\
&\leq \delta \ .
\end{aligned}$$

Here the second inequality follows since $\|(c/\theta)w-y\| \leq r$ for $(w,\theta) \in T$, the third inequality follows since $B(y,r) \subset B(0,R)$, and the last inequality follows since $\theta \leq c$ for $(w,\theta) \in T$. This shows that $T \subset B((0,0),\delta)$. Therefore $\delta^{-1}T \subset B((0,0),1)$ and hence $\delta^{-1}T \subset H$. It then follows that

$$\mathrm{Vol}(H) \geq \mathrm{Vol}(\delta^{-1}T) = \left(\frac{1}{\delta}\right)^{n+1} \int_0^c v(n)\left(\frac{\theta r}{c}\right)^n d\theta = \left(\frac{v(n)r^n}{\delta^{n+1}c^n}\right)\left(\frac{\theta^{n+1}}{n+1}\right)\Bigg|_0^c = \frac{v(n)r^n c}{\delta^{n+1}(n+1)} \ ,$$

where $v(n)$ is the volume of the n-dimensional unit ball, see (4). Therefore

$$\frac{\mathrm{vol}\left(B_{n+1}^{\bar{x}}\right)}{\mathrm{vol}\left(B_{n+1}^{\bar{x}} \cap W_c^{\bar{x}}\right)} = \frac{\mathrm{vol}\left(B_{n+1}^{\bar{x}}\right)}{\mathrm{vol}\left(H\right)} \leq \frac{v(n+1)\delta^{n+1}(n+1)}{v(n)r^n c} = \frac{\Gamma(n/2+1)\pi^{(n+1)/2}\delta^{n+1}(n+1)}{\Gamma((n+1)/2+1)\pi^{n/2}r^n c} \ .$$

We bound the right-most term involving the ratio of two gamma function values using the following inequality for the gamma function, see [16]:

$$\frac{\Gamma(x+1/2)}{\Gamma(x+1)} < \frac{1}{\sqrt{x+1/4}} \ .$$

Using $x=(n+1)/2$ in the above yields

$$\begin{aligned}
\frac{\mathrm{vol}\left(B_{n+1}^{\bar{x}}\right)}{\mathrm{vol}\left(B_{n+1}^{\bar{x}} \cap W_c^{\bar{x}}\right)} &\leq \left(\frac{1}{\sqrt{\frac{n+1}{2}+\frac{1}{4}}}\right)\frac{\pi^{(n+1)/2}\delta^{n+1}(n+1)}{\pi^{n/2}r^n c} \\
&= \frac{2(n+1)\sqrt{\pi}}{\sqrt{2n+3}}\left(\frac{\sqrt{R^2+c^2}}{r}\right)^n\left(\frac{\sqrt{R^2+c^2}}{c}\right) \\
&\leq 3.18 \cdot \sqrt{n}\left(\frac{\sqrt{R^2+c^2}}{r}\right)^n\left(\frac{\sqrt{R^2+c^2}}{c}\right) \ ,
\end{aligned}$$

where the last inequality follows from the observation that $\frac{2(n+1)}{\sqrt{n(2n+3)}} \leq 4/\sqrt{5}$ for $n \geq 1$ since the left expression is decreasing in n for $n \geq 1$. Lastly, taking logarithms yields (9).∎

We acknowledge C. Roos [17] for several constructions used in the above proof. Lemma 2.1 yields the following complexity bound for solving (1) using the extended ellipsoid method:

Theorem 2.2 *Suppose that S is given via a separation oracle, that $\bar{x} \in \Re^n$ is given, and that the extended ellipsoid algorithm is applied to solve (1). If R and r are positive scalars such that $B(\bar{x}, R) \cap S$ contains a ball of radius r, then the algorithm will solve (6), and hence solve (1), in at most*

$$\left\lceil 2(n+1)\left(\frac{1}{2}\ln(n) + 1.16 + n\ln\left(\frac{1}{r} + \frac{R}{r}\right) + \ln(1+R)\right)\right\rceil$$

iterations, where each iteration makes one call of the separation oracle for S.

Proof: The iteration bound follows immediately from Lemma 2.1 and Theorem 2.1, noting that the dimension of the space containing $W^{\bar{x}}$ is $n+1$.∎

Notice that this complexity bound requires only the specification of an initializing point $\bar{x}$, which one can think of as the "reference point." The condition "there exists R and r for which $B(\bar{x}, R) \cap S$ contains a ball of radius r" is only existential; prior knowledge of any R, r, or bounds thereon are unnecessary. The complexity bound is monotone in three quantities, namely R/r, R, and $1/r$. While it is tempting to think that R/r will be the largest of these quantities, Examples 1.1 and 1.2 of Section 1 show that R or $1/r$ might be the dominant quantity. It is also curious that among these three quantities, the complexity bound depends more weakly on R than on the other two quantities, roughly by a factor of n. We show in Section 3 that the dependence of the complexity bound on R and $1/r$ (as well as on R/r) cannot be removed. Therein we also show a weaker dependence on R than on R/r, but by a different factor than given above.

For a given point $\bar{x}$, we can say that S has favorable geometry relative to $\bar{x}$ to the extent that there exist values R and r satisfying (3) for which R/r, R, and $1/r$ are not too large. Put slightly differently, S has favorable geometry relative to $\bar{x}$ if S contains a ball whose radius r is not too small and whose distance R from $\bar{x}$ is not too large. Then Theorem 2.2 states that if S has favorable geometry relative to $\bar{x}$, then the extended ellipsoid algorithm will solve (1) relatively quickly. In Section 3 we study the converse of this statement.

Last of all, notice that the complexity bound in Theorem 2.2 is not scale-invariant, which seems unnatural at first glance. That is, one would expect if the units were changed so that both R and r were doubled, say, then the complexity bound would remain unchanged, but this is not the case. The reason for this has to do with the implicit choice of using $c = 1$ in the "lift and conify" construction used to transform S to $W_c^{\bar{x}} = W_1^{\bar{x}} = W^{\bar{x}}$ in the extended ellipsoid algorithm.

Indeed, for a given value of $c > 0$ we can implement the extended ellipsoid method described in Section 1, substituting $W^{\bar{x}}_c$ for $W^{\bar{x}}_1 = W^{\bar{x}}$. The specific choice of $c = 1$ used in the description of the extended ellipsoid algorithm is somewhat arbitrary, but the choice of c must be given, i.e., "known" to the algorithm, so that the separation oracle for S can be converted to one for $W^{\bar{x}}_c$. If we change units so that R and r are doubled, then if we double the value of c from $c = 1$ to $c = 2$ it follows from (9) that the volume ratio bound and hence the complexity bound will remain unchanged; hence the extended ellipsoid method is scale-invariant if c is re-scaled together with R and r. Furthermore, it follows from (9) that if a value of R is known in advance for which (3) is true for some $r > 0$, then setting $c = R$ in (9) yields a volume ratio bound of $O(\ln(n) + n\ln(R/r))$ and hence an iteration complexity bound of $O(n\ln(n) + n^2\ln(R/r))$, whose sole dependence on R/r (and whose independence of R and $1/r$ separately) is consistent with the complexity bound in Corollary 2.1. However, it is precisely because R is not known that the dependence on R arises (even when R/r is $O(1)$) in Lemma 2.1 and hence in the complexity bound of Theorem 2.2.

The ellipsoid algorithm belongs to a larger class of efficient volume-reducing separation-oracle based algorithms that includes the method of centers of gravity [8], the method of inscribed ellipsoids [7], and the method of volumetric centers [19], among others. Results analogous to Theorem 2.1 can be derived for these methods, for example for the method of centers of gravity the iteration complexity analogous to Theorem 2.1 is $O(\ln(n)+n\ln(1/r+R/r)+\ln(1+R))$. For a more thorough discussion of the complexity of volume-reducing cutting-plane methods, see [10].

3 Lower Bounds on Complexity of (1) for Separation Oracle Algorithms

Theorem 2.2 showed that favorable geometry implies favorable complexity: if R/r, R, and $1/r$ are all small, then the ellipsoid algorithm (and many other algorithms) will compute a solution of (1) in a small number of iterations. In this subsection we study the converse question: does favorable complexity imply favorable geometry? A naive approach to this question is as follows: supposing that a separation-oracle based algorithm solves (1) in a low number of iterations, then prove that R/r, R, and $1/r$ all must be small. This approach is obviously doomed to failure, as the algorithm could simply get lucky and compute a solution of (1) by accident at an early iteration, regardless of the values of R/r, R, and $1/r$. We therefore consider our algorithm applied not to a single instance of (1) but rather applied to all instances in certain collections of convex sets. For fixed values of $\bar{x}$, R, and r, let $\mathcal{S}^{\bar{x},r,R}$ denote the collection of convex sets $S \subset \Re^n$ whose intersection with $B(\bar{x}, R)$ contains a ball of radius r, namely:

$$\mathcal{S}^{\bar{x},r,R} = \{S \subset \Re^n : S \text{ is convex and satisfies (3) } \} \ .$$

Now consider a (separation-oracle based) algorithm $\mathcal{M}$ for solving instances of (1). (For a precise definition of a separation-oracle based algorithm see [12]; an intuitive notion of this type of algorithm is sufficient for our purposes.) An instance is a given convex set S, or more precisely, a separation oracle for the convex set S. Now suppose we fix $\bar{x}$, R, and r, and restrict our instances to (separation oracles for) sets S in the collection $\mathcal{S}^{\bar{x},r,R}$. Let $N^{\bar{x},r,R}(\mathcal{M})$ denote the computational complexity of algorithm $\mathcal{M}$ over all instances $S \in \mathcal{S}^{\bar{x},r,R}$. That is, $N^{\bar{x},r,R}(\mathcal{M})$ is the maximum number of oracle calls it takes the algorithm $\mathcal{M}$ to solve (1) over all (separation oracles for) sets S in the collection $\mathcal{S}^{\bar{x},r,R}$. Our first lower bound result is as follows:

Theorem 3.1 *For any fixed $\bar{x}$, R, and r satisfying $R \geq r > 0$, let $\mathcal{M}$ be any separation-oracle based algorithm applied over the collection of sets $\mathcal{S}^{\bar{x},r,R}$. Then*

$$N^{\bar{x},r,R}(\mathcal{M}) \geq \left\lfloor \log_2\left(\frac{R}{r}\right) - 1 \right\rfloor \ .$$

Proof: We use a type of argument used extensively in [12] that works by constructing output of a separation oracle for a particular set $\bar{S} \in \mathcal{S}^{\bar{x},r,R}$ for which the algorithm makes at least $\left\lfloor \log_2\left(\frac{R}{r}\right) - 1 \right\rfloor$ oracle calls. Without loss of generality we presume that $\bar{x} = 0$, which will lead to simpler arithmetic in the proof. Let x^1 be the first point used to query the separation oracle. (This point is generated by the algorithm independent of any information from the separation oracle or, equivalently, the set in question.) Let $L_0 = -R$ and $U_0 = R$. If $(x^1)_1 \leq \frac{L_0+U_0}{2}$, the oracle will return "$x^1 \notin \bar{S}$" together with the separating halfspace $H^{++} := \{x \in \Re^n : x_1 > \frac{L_0+U_0}{2}\}$ for which $\bar{S} \subset H^{++}$. Henceforth in this proof and other proofs we simply denote this as "$\bar{S} \subset \{x \in \Re^n : x_1 > \frac{L_0+U_0}{2}\}$." If instead $(x^1)_1 > \frac{L_0+U_0}{2}$, the oracle will return "$x^1 \notin \bar{S}$" together with "$\bar{S} \subset \{x \in \Re^n : x_1 < \frac{L_0+U_0}{2}\}$." In the first case we will define $L_1 := \frac{L_0+U_0}{2}$ and $U_1 := U_0$, whereas in the second case we define $L_1 := L_0$ and $U_1 := \frac{L_0+U_0}{2}$. We will construct the output of the separation oracle in subsequent iterations in a manner that generalizes the above logic. After k oracle calls we will have two scalar values L_k and U_k satisfying $L_k < U_k$, and the algorithm will have generated $x^1, \ldots, x^k$ for which the oracle has responded "$x^i \notin \bar{S}$" together with separating halfspaces of the form "$\bar{S} \subset \{x \in \Re^n : x_1 > \text{(or } <) \ \frac{L_{i-1}+U_{i-1}}{2}\}$" (depending on the position of $(x^i)_1$) for $i = 1, \ldots, k$. The algorithm will next generate x^{k+1} and query the oracle with this point. If $(x^{k+1})_1 \leq \frac{L_k+U_k}{2}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_{k+1} > \frac{L_k+U_k}{2}\}$." If instead $(x^{k+1})_1 > \frac{L_k+U_k}{2}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 < \frac{L_k+U_k}{2}\}$." In the first case we will define $L_{k+1} := \frac{L_k+U_k}{2}$ and $U_{k+1} := U_k$, whereas in the second case we define $L_{k+1} := L_k$ and $U_{k+1} := \frac{L_k+U_k}{2}$. We proceed inductively until the algorithm has made $K = \lfloor \log_2(R/r) - 1 \rfloor$ oracle calls (iterations) and we have generated a (monotone increasing) sequence $\{L_i\}_{i=0}^K$ and a (monotone decreasing) sequence $\{U_i\}_{i=0}^K$ according to the above rules.

Now define

$$\bar{S} = \{x \in \Re^n : \|x\| \leq R,\ L_K + \delta \leq x_1 \leq U_K - \delta\}$$

where $\delta := R \cdot 2^{-(K+1)}$. Then it follows that the separating hyperplanes generated from the oracle calls are consistent with the instance $\bar{S}$. We first argue that the points $x^1, \ldots, x^K \notin \bar{S}$. To see this define $C^i := \{x \in \Re^n : \|x\| \leq R \text{ and } L_i + \delta \leq x_1 \leq U_i - \delta\}$ for $i = 1, \ldots, K$. Then it follows that $x^i \notin C^i$ and $\bar{S} \subset C^K \subset \cdots \subset C^2 \subset C^1$, therefore $x^i \notin \bar{S}$ for $i = 1, \ldots, K$. We claim that $\bar{S} \in \mathcal{S}^{\bar{x},r,R}$. Notice trivially that $\bar{S} \subset B(\bar{x}, R)$ (recall that $\bar{x} = 0$), so it remains to prove that $\bar{S}$ contains a ball of radius r. To see this, notice that $U_i - L_i = 2R2^{-i}$ for all $i = 0, \ldots, K$, and observe that the upper/lower bounds on the first coordinate of points in $\bar{S}$ satisfy:

$$\frac{U_K - \delta - (L_K + \delta)}{2} = \frac{U_K - L_K}{2} - R \cdot 2^{-(K+1)} = R \cdot 2^{-(K)} - R \cdot 2^{-(K+1)} = R \cdot 2^{-(K+1)} \geq r\ ,$$

from which it easily follows that $B(y, r) \subset \bar{S}$ for $y = \frac{U_K + L_K}{2}(1, 0, \ldots, 0)$. Hence $\bar{S} \in \mathcal{S}^{\bar{x},r,R}$ and by construction, $x^i \notin \bar{S}$, $i = 1, \ldots, K$. Therefore the algorithm makes at least K oracle calls, proving the result. ▮

The general technique used in the above proof (and which will also be used to prove the other two lower bound theorems in this section) was borrowed from [12]. Notice that it involves inductively using the output of the given algorithm $\mathcal{M}$ to create a "resisting (separation) oracle" for a particular (algorithm dependent) set $\bar{S} \in \mathcal{S}^{\bar{x},r,R}$, in such a way that the algorithm must make at least a certain number of oracle calls. (The appelation "resisting oracle" was aptly introduced by Nesterov in [13].) In the above proof, which is essentially unidimensional, the resisting oracle is constructed in such a way that for at least K oracle calls, the algorithm $\mathcal{M}$ generates points x^i which are not contained in the instance $\bar{S}$. This approach will be modified in the proofs of the two additional lower bounds of Theorems 3.2 and 3.3.

The method of centers of gravity [8] can be shown to achieve the lower bound in Theorem 3.1; the proof of this result can be gleaned from similar arguments in [12].

We next show that the lower bound complexity also depends monotonically on R even when R/r is bounded above by a constant. Analogous to the family of sets $\mathcal{S}^{\bar{x},r,R}$, we will need a suitable family of (separation oracles for) convex sets that will serve as instances for applying any algorithm $\mathcal{M}$ to solve (1). For fixed values of $\bar{x}$, R, and aspect ratio bound A, let $\mathcal{T}^{\bar{x},R,A}$ denote the collection of convex sets $S \subset \Re^n$ that satisfy:
(i) there exists $\bar{R}$ and $\bar{r}$ for which $B(\bar{x}, \bar{R}) \cap S$ contains a ball of radius $\bar{r}$,
(ii) $\bar{R} \leq R$, and
(iii) $\bar{R}/\bar{r} \leq A$.

For a given separation-oracle based algorithm $\mathcal{M}$, let $N^{\bar{x},R,A}(\mathcal{M})$ denote the computational complexity of algorithm $\mathcal{M}$ over all instances $S \in \mathcal{T}^{\bar{x},R,A}$. That is, $N^{\bar{x},R,A}(\mathcal{M})$ is the maximum number of oracle calls it takes the algorithm $\mathcal{M}$ to solve (1) over all (separation oracles for) sets S in the collection $\mathcal{T}^{\bar{x},R,A}$. We have:

Theorem 3.2 *For any fixed $\bar{x}$, R, and A satisfying $R \geq 1$ and $A \geq 4$, let $\mathcal{M}$ be any separation-oracle based algorithm applied over the collection of sets $\mathcal{T}^{\bar{x},R,A}$. Then*

$$N^{\bar{x},R,A}(\mathcal{M}) \geq \lfloor \log_2 \log_2 (R+1) \rfloor \ .$$

We offer the following interpretation of this theorem before proving it. For any given algorithm $\mathcal{M}$, there exists a (separation oracle for a) convex set $\bar{S} \in \mathcal{T}^{\bar{x},R,A}$ for which the iteration complexity of the algorithm grows at least as $\log_2 \log_2(R)$, independent of $1/r$ or the aspect ratio A of $\bar{S}$, provided that $R \geq 1$ and $A \geq 4$.

Proof: The proof uses ideas communicated privately from Nemirovsky [11], and constructs a resisting oracle for a particular set $\bar{S} \in \mathcal{T}^{\bar{x},R,A}$. However, unlike the "binary evasion" strategy used to prove Theorem 3.1, in this proof the evasion takes place on an exponential scale, using intervals of the form $[2^{L_i},\ 2^{U_i}]$ as opposed to $[L_i,\ U_i]$. Without loss of generality we presume that $\bar{x} = e^1 := (1, 0, \ldots, 0)$, which will lead to simpler arithmetic in the proof. Let x^1 be the first point used to query the separation oracle. (This point is generated by the algorithm independent of any information from the separation oracle or, equivalently, the set in question.) Let $K = \lfloor \log_2 \log_2 (1+R) \rfloor$, and define $L_0 = 0$ and $U_0 = 2^K$. We will construct the separation oracle inductively as follows. If $(x^1)_1 \leq 2^{\frac{L_0+U_0}{2}}$, the oracle will return "$x^1 \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 > 2^{\frac{L_0+U_0}{2}}\}$." If instead $(x^1)_1 > 2^{\frac{L_0+U_0}{2}}$, the oracle will return "$x^1 \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 < 2^{\frac{L_0+U_0}{2}}\}$." In the first case we will define $L_1 := \frac{L_0+U_0}{2}$ and $U_1 := U_0$, whereas in the second case we define $L_1 := L_0$ and $U_1 := \frac{L_0+U_0}{2}$. On subsequent iterations we will construct the output of the separation oracle in a manner that generalizes the above logic. After k oracle calls we will have two scalar values L_k and U_k satisfying $L_k \leq U_k$, and the algorithm will have generated $x^1, \ldots, x^k$ for which the oracle has responded "$x^i \notin \bar{S}$" together with separating halfspaces of the form "$\bar{S} \subset \{x \in \Re^n : x_1 >$ (or $<$) $2^{\frac{L_{i-1}+U_{i-1}}{2}}\}$" (depending on the position of $(x^i)_1$) for $i = 1, \ldots, k$. The algorithm will next generate x^{k+1} and query the oracle with this point. If $(x^{k+1})_1 \leq 2^{\frac{L_k+U_k}{2}}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_{k+1} > 2^{\frac{L_k+U_k}{2}}\}$." If instead $(x^{k+1})_1 > 2^{\frac{L_k+U_k}{2}}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 < 2^{\frac{L_k+U_k}{2}}\}$." In the first case we will define $L_{k+1} := \frac{L_k+U_k}{2}$ and $U_{k+1} := U_k$, whereas in the second case we define $L_{k+1} := L_k$ and $U_{k+1} := \frac{L_k+U_k}{2}$. We proceed iteratively until the algorithm has made K oracle calls (iterations) and we have generated a (monotone increasing) sequence $\{L_i\}_{i=0}^K$ and a (monotone decreasing) sequence $\{U_i\}_{i=0}^K$ according to the above rules.

Now define the following objects:

$$\begin{array}{llllll} \delta & := 1/4 & \bar{r} & := \frac{2^{U_K}-2^{L_K}}{2} - \delta & \bar{R} & := 2^{U_K} - 1 \\ \bar{y} & := \frac{2^{L_K}+2^{U_K}}{2} e^1 & \bar{S} & := \{x \in \Re^n : \|x - \bar{y}\| \le \bar{r}\} \ . \end{array}$$

Then it follows that the separating hyperplanes that were the output after each oracle are consistent with the instance $\bar{S}$. We first argue that the points $x^1, \ldots, x^K \notin \bar{S}$. To see this define $C^i := \{x \in \Re^n : 2^{L_i} + \delta \le x_1 \le 2^{U_i} - \delta\}$ for $i = 1, \ldots, K$. Then it follows that $x^i \notin C^i$ and $\bar{S} \subset C^K \subset \cdots \subset C^2 \subset C^1$, therefore $x^i \notin \bar{S}$ for $i = 1, \ldots, K$.

We claim that $\bar{S} \in \mathcal{T}^{\bar{x},R,A}$. We first prove that $\bar{S} \subset B(\bar{x}, \bar{R}) \subset B(\bar{x}, R)$. Indeed, let $x \in \bar{S}$, then noticing that $U_i - L_i = 2^{K-i}$ for all $i = 0, \ldots, K$ whereby $U_K - L_K = 1$, and we have:

$$\|x-\bar{x}\| \le \|x-\bar{y}\|+\|\bar{y}-\bar{x}\| \le \bar{r}+\frac{2^{L_K}+2^{U_K}}{2}-1 = 2^{U_K}-\delta-1 < \bar{R} \le 2^{U_0}-1 = 2^{2^K}-1 \le R+1-1 = R \, ,$$

thus showing that $x \in B(\bar{x}, \bar{R}) \subset B(\bar{x}, R)$ and proving this first part of the claim.

We complete the proof by showing that $\bar{R}/\bar{r} \le A$. First observe:

$$\bar{r} = \frac{2^{U_K} - 2^{L_K}}{2} - \delta = \frac{2^{L_K+1} - 2^{L_K}}{2} - \delta = \frac{2^{L_K}}{2} - 1/4 \ge \frac{2^{L_0}}{2} - 1/4 = \frac{1}{2} - \delta = 1/4 \ ,$$

from which it follows that:

$$\frac{\bar{R}}{\bar{r}} = \frac{2^{U_K} - 1}{\frac{2^{U_K}-2^{L_K}}{2} - \delta} = \frac{2^{L_K+1} - 1}{2^{L_K-1} - \delta} = \frac{2^{L_K+1} - 1}{2^{L_K-1} - 1/4} = 4 \le A \ .$$

Hence $\bar{S} \in \mathcal{T}^{\bar{x},R,A}$ and by construction, $x^i \notin \bar{S}$, $i = 1, \ldots, K$. Therefore the algorithm makes at least K oracle calls, proving the result. ▌

Remark 3.1 *Note in the proof of Theorem 3.2 that the instance $\bar{S}$ constructed in the proof has an aspect ratio bounded above by 4, and also satisfies $\bar{r} \ge 1/4$. Therefore we could equivalently re-phrase Theorem 3.2 to state that $A = 4$ rather than $A \ge 4$.*

Last of all, we show that the lower bound complexity also depends monotonically on $1/r$ even when R/r is small. Analogous to the family of sets $\mathcal{T}^{\bar{x},R,A}$, we will need a suitable family of (separation oracles for) convex sets that will serve as instances for applying any algorithm $\mathcal{M}$ to solve (1). For fixed values of $\bar{x}$, r, and aspect ratio bound A, let $\mathcal{U}^{\bar{x},r,A}$ denote the collection of convex sets $S \subset \Re^n$ that satisfy:

(i) there exists $\bar{R}$ and $\bar{r}$ for which $B(\bar{x}, \bar{R}) \cap S$ contains a ball of radius $\bar{r}$,

(ii) $\bar{r} \geq r$, and

(iii) $\bar{R}/\bar{r} \leq A$.

For a given separation-oracle based algorithm $\mathcal{M}$, let $N^{\bar{x},r,A}(\mathcal{M})$ denote the computational complexity of algorithm $\mathcal{M}$ over all instances $S \in \mathcal{U}^{\bar{x},r,A}$. That is, $N^{\bar{x},r,A}(\mathcal{M})$ is the maximum number of oracle calls it takes the algorithm $\mathcal{M}$ to solve (1) over all (separation oracles for) sets S in the collection $\mathcal{U}^{\bar{x},r,A}$. We have:

Theorem 3.3 *For any fixed $\bar{x}$, r, and A satisfying $0 < r < 1/4$, and $A \geq 4$, let $\mathcal{M}$ be any separation-oracle based algorithm applied over the collection of sets $\mathcal{U}^{\bar{x},r,A}$. Then*

$$N^{\bar{x},r,A}(\mathcal{M}) \geq \left\lfloor \log_2 \log_2 \left(\frac{1}{4r} \right) \right\rfloor \ .$$

We offer the following interpretation of this theorem before proving it. For any given algorithm $\mathcal{M}$, there exists a (separation oracle for a) convex set $\bar{S} \in \mathcal{U}^{\bar{x},r,A}$ for which the iteration complexity of the algorithm grows at least as $\log_2 \log_2(1/r)$, independent of R or the aspect ratio A of $\bar{S}$, provided that $r < 1/4$ and $A \geq 4$.

Proof: The proof is similar in structure to that of Theorem 3.2, except that instead of exponentially scaled intervals $[2^{L_i},\ 2^{U_i}]$ we use inverse exponentially scaled intervals $[2^{-L_i},\ 2^{-U_i}]$, and some other arithmetic is different as well. Let $K = \lfloor \log_2 \log_2(1/(4r)) \rfloor$, $L_0 := 2^K$, $U_0 := 0$, and without loss of generality we presume that $\bar{x} = (1/2)^{L_0} e^1$, which will lead to simpler arithmetic in the proof.

Let x^1 be the first point used to query the separation oracle. (This point is generated by the algorithm independent of any information from the separation oracle or, equivalently, the set in question.) We will construct the separation oracle inductively as follows. If $(x^1)_1 \leq (1/2)^{\frac{L_0+U_0}{2}}$, the oracle will return "$x^1 \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 > (1/2)^{\frac{L_0+U_0}{2}}\}$." If instead $(x^1)_1 > (1/2)^{\frac{L_0+U_0}{2}}$, the oracle will return "$x^1 \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 < (1/2)^{\frac{L_0+U_0}{2}}\}$." In the first case we will define $L_1 := \frac{L_0+U_0}{2}$ and $U_1 := U_0$, whereas in the second case we define $L_1 := L_0$ and $U_1 := \frac{L_0+U_0}{2}$. On subsequent iterations we will construct the output of the separation oracle in a manner that generalizes the above logic. After k oracle calls we will have two scalar values L_k and U_k satisfying $L_k > U_k$, and the algorithm will have generated $x^1, \ldots, x^k$ for which the oracle has responded "$x^i \notin \bar{S}$" together with separating halfspaces of the form "$\bar{S} \subset \{x \in \Re^n : x_1 >$ (or $<$) $2^{\frac{L_{i-1}+U_{i-1}}{2}}\}$" (depending on the position of $(x^i)_1$) for $i = 1, \ldots, k$. The algorithm will next generate x^{k+1} and query the oracle with this point. If $(x^{k+1})_1 \leq (1/2)^{\frac{L_k+U_k}{2}}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_{k+1} > (1/2)^{\frac{L_k+U_k}{2}}\}$." If instead $(x^{k+1})_1 > (1/2)^{\frac{L_k+U_k}{2}}$, the oracle will return "$x^{k+1} \notin \bar{S}$" together with the separating halfspace "$\bar{S} \subset \{x \in \Re^n : x_1 < (1/2)^{\frac{L_k+U_k}{2}}\}$." In the first case we will define $L_{k+1} := \frac{L_k+U_k}{2}$ and $U_{k+1} := U_k$, whereas in the second case we define $L_{k+1} := L_k$ and $U_{k+1} := \frac{L_k+U_k}{2}$. We proceed iteratively until the algorithm has made K oracle calls (iterations) and we have generated a (monotone decreasing) sequence $\{L_i\}_{i=0}^K$ and a (monotone increasing) sequence $\{U_i\}_{i=0}^K$ according to the above rules.

Now define the following objects:

$$\begin{array}{llllllll}
\delta & := & \min\left\{\frac{1}{4}\left(\frac{1}{2}\right)^{L_0}, r\right\} & \bar{r} & := & \frac{(1/2)^{U_K}-(1/2)^{L_K}}{2} - \delta & \bar{R} & := & (1/2)^{U_K} - (1/2)^{L_0} \\
\bar{y} & := & \frac{(1/2)^{L_K}+(1/2)^{U_K}}{2}e^1 & \bar{S} & := & \{x \in \Re^n : \|x - \bar{y}\| \le \bar{r}\} \ .
\end{array}$$

Then it follows that the separating hyperplanes that were the output after each oracle are consistent with the instance $\bar{S}$. Note that $\bar{R} > 0$; to see this observe that $L_i - U_i = 2^{K-i}$ for all $i = 0, \ldots, K$, whereby $L_K - U_K = 1$ and:

$$\bar{R} = \left(\frac{1}{2}\right)^{L_K-1} - \left(\frac{1}{2}\right)^{L_0} = \left(\frac{1}{2}\right)^{L_K}\left(2 - \left(\frac{1}{2}\right)^{L_0-L_K}\right) > 0 \ .$$

We first argue that the points $x^1, \ldots, x^K \notin \bar{S}$. To see this define $C^i := \{x \in \Re^n : (1/2)^{L_i} + \delta \le x_1 \le (1/2)^{U_i} - \delta\}$ for $i = 1, \ldots, K$. Then it follows that $x^i \notin C^i$ and $\bar{S} \subset C^K \subset \cdots \subset C^2 \subset C^1$, therefore $x^i \notin \bar{S}$ for $i = 1, \ldots, K$.

We claim that $\bar{S} \in \mathcal{U}^{\bar{x},r,A}$. We first prove that $\bar{S} \subset B(\bar{x}, \bar{R})$. Indeed, let $x \in \bar{S}$, then we have:

$$\|x - \bar{x}\| \le \|x - \bar{y}\| + \|\bar{y} - \bar{x}\| \le \bar{r} + \frac{(1/2)^{L_K} + (1/2)^{U_K}}{2} - (1/2)^{L_0} = (1/2)^{U_K} - (1/2)^{L_0} - \delta < \bar{R} \ ,$$

thus showing that $x \in B(\bar{x}, \bar{R})$ and proving this first part of the claim.

We next show that $\bar{r} \ge r$. We have:

$$\bar{r} = \frac{(1/2)^{L_K-1} - (1/2)^{L_K}}{2} - \delta = \frac{1}{2}\left(\frac{1}{2}\right)^{L_K} - \delta \ge \frac{1}{2}\left(\frac{1}{2}\right)^{L_0} - \delta = \frac{1}{2}\left(\frac{1}{2^{2^K}}\right) - \delta \ge \frac{4r}{2} - \delta = 2r - \delta \ge r \ .$$

Last of all, we show that $\bar{R}/\bar{r} \le A$, which will complete the proof. We have:

$$\frac{\bar{R}}{\bar{r}} = \frac{(1/2)^{U_K} - (1/2)^{L_0}}{\frac{1}{2}\left(\frac{1}{2}\right)^{L_K} - \delta} \le \frac{2(1/2)^{L_K} - (1/2)^{L_0}}{\frac{1}{2}\left(\frac{1}{2}\right)^{L_K} - \frac{1}{4}\left(\frac{1}{2}\right)^{L_0}} = 4 \le A \ .$$

Hence $\bar{S} \in \mathcal{U}^{\bar{x},r,A}$ and by construction, $x^i \notin \bar{S}$, $i = 1, \ldots, K$. Therefore the algorithm makes at least K oracle calls, proving the result. ▌

Remark 3.2 *Note, just as in the case of Theorem 3.2, that in the proof of Theorem 3.3 that the instance $\bar{S}$ constructed in the proof has an aspect ratio bounded above by 4. (Also notice in the proof that $\bar{R} \leq 1$.) Therefore we could equivalently re-phrase the theorem to state that $A = 4$ rather than $A \geq 4$.*

4 Discussion and Further Questions

4.1 On Stronger Lower Bounds

There is a significant gap between the lower bound results in Theorems 3.1, 3.2, and 3.3 and the upper bound result in Theorem 2.2. One reason for the gap has to do with the absence of any dimensional factor n in the lower bound results. This is partially an artifact of the proof constructions in the lower bound theorems, which are all essentially unidimensional in nature. It might be of interest to strengthen the lower bound theorems by taking explicit advantage of the dimension n in constructing suitable resisting oracles with stronger lower bounds. But even when $n = 1$ there is a gap between the lower bound and the upper bound results concerning the dependence on R and $1/r$ in Theorems 3.2 and 3.3, by a logarithmic factor. It would also be interesting to strengthen these lower bound results by removing the extra logarithmic factor in the lower bound results.

4.2 Lower Bounds on Complexity for the (self-concordant) Barrier Model

Taken together, Theorems 3.1, 3.2, and 3.3 show that there exist problem instances in certain families characterized by unfavorable geometry, that require more computational effort to solve by any separation oracle algorithm. This naturally begs the question whether such an implication might extend to the case where the set is described instead by a self-concordant barrier. In particular, consider the case when S is presented in conic format (2) and the cone K has a computable self-concordant barrier. It is shown in [5] that in this case favorable geometry of S implies favorable computational complexity of a suitably defined barrier method for solving (1). However, it is an open challenge to prove an assertion that unfavorable geometry implies (say, in the worst case) a large number of iterations of any (self-concordant) barrier method.

Acknowledgement. We are grateful to two anonymous referees for their suggestions on ways to improve the paper. The second author would also like to acknowledge the support of the MIT

Sloan School of Management as this research was carried out while being a visiting professor there.

References

[1] A. Belloni and R. Freund. A geometric analysis of Renegar's condition number, and its interplay with conic curvature. *Mathematical Programming*, to appear, 2008.

[2] R. Bland, D. Goldfarb, and Michael J. Todd. The ellipsoid method: a survey. *Operations Research*, 29(6):1039–1091, 1981.

[3] R. M. Freund and J. R. Vera. Condition-based complexity of convex optimization in conic linear form via the ellipsoid algorithm. *SIAM Journal on Optimization*, 10(1):155–176, 1999.

[4] R. M. Freund and J. R. Vera. Some characterizations and properties of the "distance to ill-posedness" and the condition measure of a conic linear system. *Mathematical Programming*, 86(2):225–260, 1999.

[5] Robert M. Freund. Complexity of convex optimization using geometry-based measures and a reference point. *Mathematical Programming*, 99:197–221, 2004.

[6] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, Berlin, 1998.

[7] L.G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Math. Dokl.*, 20(1):191–194, 1979.

[8] A.L. Levin. On an algorithm for the minimization of convex functions. *Soviet Mathematics Doklady*, 6:286–290, 1965.

[9] L.Porkolab and L.Khachiyan. On the complexity of semidefinite programs. *Journal of Global Optimization*, 10:351–365, 1997.

[10] T.L. Magnanti and G. Perakis. A unifying geometric solution framework and complexity analysis for variational inequalities. *Mathematical Programming*, 71:327–351, 1995.

[11] A. Nemirovsky. private communication. 1998.

[12] A.S. Nemirovsky and D.B. Yudin. Informational complexity and efficient methods for solving complex extremal problems. *Ekonomika i Matem. Metody*, 12:357–369, 1976.

[13] Yuri Nesterov. *Introductory Lectures on Convex Optimization.* Kluwer Academic Publishers, Norwell, MA, 2004.

[14] J. Renegar. Some perturbation theory for linear programming. *Mathematical Programming*, 65(1):73–91, 1994.

[15] J. Renegar. Linear programming, complexity theory, and elementary functional analysis. *Mathematical Programming*, 70(3):279–351, 1995.

[16] Wolfram Research. Gamma function: Inequalities. `http://functions.wolfram.com/GammaBetaErf/Gamm` June 2008.

[17] C. Roos. private communication. 1999.

[18] F. Rosenblatt. *Principles of Neurodynamics.* Spartan Books, Washington, DC, 1962.

[19] P. Vaidya. A new algorithm for minimizing convex functions over convex sets. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 338–343, Los Alamitos, CA, 1989. IEEE Computer Soc. Press.