Finding the Needle in a High-Dimensional Haystack: Oracle Methods for Convex Optimization

Tyler LaBonte
Advisor: Prof. Shaddin Dughmi

Senior Thesis
University of Southern California
Viterbi School of Engineering
Department of Computer Science
Spring 2021

Abstract

Convex optimization and linear programming are fundamental, powerful techniques in mathematical optimization. Under certain conditions, including the presence of a separation oracle for the feasible region, the Ellipsoid Method is a polynomial-time algorithm for solving convex optimization problems. Decades of work on cutting-plane methods have resulted in Ellipsoid-like algorithms which achieve optimal oracle complexity in this setting.

However, obtaining an efficient separation oracle is not always possible or natural. Recent work has studied membership, approximation, index, and quantum oracles as alternatives to separation. While each is theoretically insightful, only quantum oracles have improved upon the oracle complexity of cutting-plane methods.

We propose a classical alternative oracle, the distance oracle, which returns the minimum Euclidean distance from the query point to the feasible region. We present progress on an algorithm for solving the linear feasibility problem with a distance oracle in $O(n \log(n))$ queries, a logarithmic improvement in oracle complexity over the best separation oracle-based method. For problems where distance oracles are natural, such as interactive learning, this enables an efficient algorithm for optimization. We also formalize a framework for analyzing oracle strength and develop an oracle power hierarchy; as a consequence, we show that the distance oracle is stronger than the separation oracle.

Acknowledgments

This work would not have been possible without the support of many, many people throughout my undergraduate career. Thanks to Prof. Michael Shindler for getting me interested in algorithms and theoretical computer science. Thanks to Prof. David Kempe for patiently advising me on my first research project, teaching an inspiring CSCI 670 and CSCI 671, suggesting applications for this work, lending me textbooks, and overall being a great resource for all of my questions. Thanks to Prof. Shaddin Dughmi for challenging me to grow in my research ability, advising this project with enthusiasm, teaching a phenomenal CSCI 672 and CSCI 675, and encouraging me to view research problems from first principles. These three did more than anyone else to convince me that I want a research career.

Thanks to USC Viterbi for the tuition and funding, which enabled me to realistically financially consider graduate school. Thanks to Justin Fletcher, Cari Martinez, Scott Roberts, and Dan Silva for incredible internship experiences.

Thanks to all my fellow students who supported me at one time or another, in no particular order: Drew Charters, Ishan Shah, Grant Stenger, Nihar Sheth, Max Newman, Ben Brooks, Chris Hailey, Chris Fucci, Isaac Gelman, Paul Kaster, Ted Lewitt, Nat Redfern, Kian Ghodoussi, Priyank Aranke, Leena Mathur, Jillian Khoo, Stephanie Lampotang, Davey Tanaka, Jonah Yamato, Max Dykeman, Matt Ferland, Curtis Bechtel, and Sami Abu-El-Haija. Thanks especially to Max Daniels for suggesting the triangulation method for distance oracles. Apologies to anyone I left out.

Lastly, thanks to Devyn, my better half and my biggest supporter, without whom I wouldn't be where I am today. This one's for you.

Contents

Introduction

Convex optimization and linear programming are two of the most ubiquitous mathematical optimization frameworks, with myriad applications to economics [21], algorithm design and graph theory [18], machine learning and statistical estimation [8], and geometric problems [8]. In addition to its wide applicability, the study of convex optimization has generated deep theoretical implications for complexity theory [18], convex analysis [18], and game theory [14]. Today, convex optimization remains an active area of research, with recent algorithms matching conjectured runtime lower bounds [21] and taking advantage of breakthroughs in quantum computing [3, 10].

The first major algorithmic result in convex optimization was Shor, Yudin, Nemirovskii, and Khachiyan's Ellipsoid Method [32, 37, 23]. This algorithm showed that linear programming is solvable in polynomial time, even for problems with exponentially many constraints, as long as one could obtain a *separation oracle* for the feasible polytope. This discovery ignited an explosion of work in the field, leading to efficient algorithms for general convex optimization and a theoretical characterization of fundamental topics in optimization. The crown jewel of this research is the polynomial-time equivalence of separation and optimization: any convex set which admits an efficient (*i.e.,* polynomial-time) separation oracle also admits an efficient optimization algorithm, and vice versa [18]. This deep theoretical result showed that convex optimization is polynomial-time equivalent to finding a violated constraint for any proposed point.

However, a separation oracle is not always easy to obtain. It is oftentimes $\mathcal{NP}$-hard to solve the separation problem, as in fractional coloring [20]. Or perhaps the real-world con-

straints of the optimization problem prevent it, such as in power control [5]. Furthermore, it may be that separation oracles are inefficient compared to other choices of oracle that may be more suited to the problem structure – similarly to how sorting with a comparison oracle is $\Omega(n \log(n))$ but radix sort is $O(Wn)$ for key length W.

Consequently, alternative oracle methods for convex optimization have gained much attention. In addition to developing faster separation oracle algorithms [24, 33, 4, 6, 26, 21], researchers have studied membership and index oracles to probe the power of weaker oracles [6, 22, 1, 25, 5], approximate oracles to account for when the separation problem is hard [20, 16, 34], and quantum oracles to obtain faster algorithms [3, 10]. While each oracle method is insightful in its own way, only quantum oracles have resulted in optimization speedups. And, though individual oracle reductions have been well-studied, there is at best a haphazard theoretical framework for understanding oracle power relationships.

This original work component of this thesis studies the following two questions:

1. *Are there any classical oracles which provide optimization speedups over separation oracles, while being natural to implement?*

2. *How can we characterize the tradeoff between quality of oracle information and the complexity of solving the feasibility problem?*

We propose a novel type of oracle called the distance oracle, which returns the minimum Euclidean distance from the query point to the feasible region. We show progress on an algorithm for solving the linear feasibility problem with the distance oracle in $O(n \log(n))$ queries, a logarithmic improvement over the best separation oracle-based method. The convex feasibility problem is slightly more difficult, but we conjecture that we can also solve the weak problem (*i.e.,* find a δ-approximately feasible point) in $O(n \log(\frac{n}{\delta}))$ queries. We describe scenarios where distance oracles are natural, such as interactive learning, and show how the distance oracle enables an efficient algorithm for optimization.

In addition to algorithms involving the distance oracle, we are particularly interested in its relationship to other common oracles. We develop a framework for analyzing oracle

power and show how recent work in oracle methods fits into our system; in particular, we show that the distance oracle is stronger than the separation oracle. Finally, we describe some open questions in the study of oracle methods, whose answers would increase the comprehensiveness of our work.

How to Read this Thesis

This thesis is organized into three chapters. The first chapter provides a broad background of convex optimization, the Ellipsoid Method, and some important theorems regarding separation and membership oracles. This material is mainly distilled from Grötschel, Lóvasz, and Schrijver's classic textbook on geometric algorithms [18], Boyd and Vandenberghe's textbook on convex optimization [8], Sébastien Bubeck's recent survey [9], and Shaddin Dughmi's graduate convex optimization course at USC [13, 15]. This chapter can be skipped if the reader has an advanced undergraduate or beginning graduate-level understanding of convex optimization.

The second chapter is a survey of recent works in developing efficient oracle algorithms for convex optimization. The first section studies cutting-plane methods, cousins of the Ellipsoid Method, and their vast improvements in oracle complexity and runtime. The second section covers work in alternative oracles which can be used to efficiently optimize over convex sets without a separation oracle.

The third chapter is composed of original work. We introduce the distance oracle and its applications, then show progress on an algorithm for solving the linear feasibility problem with $O(n \log(n))$ oracle complexity. Then, we formalize a framework for comparing oracle power and describe some consequences of our system.

Chapter 1

Convex Optimization

1.1 Background

1.1.1 Optimization Classes

A mathematical optimization problem has the form

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && f_i(\boldsymbol{x}) \leq b_i, \quad i = 1, \ldots, m. \end{aligned} \tag{1.1}$$

Here, $\boldsymbol{x} \in \mathbb{R}^n$ is the decision variable, $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, and the $f_i : \mathbb{R}^n \to \mathbb{R}$ are the constraint functions. The set $\mathcal{X} \subset \mathbb{R}^n$ formed by the constraints is called the feasible set; it may be empty. A point $\boldsymbol{x}^\star \in \mathcal{X}$ is called locally optimal if there exists an open ball $B \subseteq \mathbb{R}^n$ centered at $\boldsymbol{x}^\star$ such that $f(\boldsymbol{x}^\star) \leq f(\boldsymbol{y})$ for all $\boldsymbol{y} \in B \cap \mathcal{X}$ and globally optimal if $f(\boldsymbol{x}^\star) \leq f(\boldsymbol{y})$ for all $\boldsymbol{y} \in \mathcal{X}$.

In general, such problems are intractable to solve in polynomial time. One class that can be solved efficiently is convex optimization problems, which have convex objective and constraint functions:

$$f_i(\alpha \boldsymbol{x} + (1 - \alpha)\boldsymbol{y}) \leq \alpha f_i(\boldsymbol{x}) + (1 - \alpha) f_i(\boldsymbol{y}) \tag{1.2}$$

for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and all $\alpha \in [0, 1]$. Therefore, the feasible set is also convex:

$$\alpha\boldsymbol{x} + (1 - \alpha)\boldsymbol{y} \in \mathcal{X} \tag{1.3}$$

for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$. If $\mathcal{X}$ is additionally compact with nonempty interior, it is called a convex body. Convex optimization problems are particularly useful and elegant because of the following fact:

Fact 1. *Every locally optimal solution to a convex optimization problem is globally optimal.*

Proof. Let $\boldsymbol{x}^\star, \boldsymbol{y} \in \mathcal{X}$ with $\boldsymbol{x}^\star$ locally optimal. By local optimality, there exists θ such that

$$f(\boldsymbol{x}^\star) \leq f(\theta\boldsymbol{x}^\star + (1 - \theta)\boldsymbol{y}). \tag{1.4}$$

By convexity of $\mathcal{X}$, $\theta\boldsymbol{x}^\star + (1 - \theta)\boldsymbol{y}$ is feasible. By convexity of f,

$$f(\theta\boldsymbol{x}^\star + (1 - \theta)\boldsymbol{y}) \leq \theta f(\boldsymbol{x}^\star) + (1 - \theta)f(\boldsymbol{y}). \tag{1.5}$$

Thus $f(\boldsymbol{x}^\star) \leq f(\boldsymbol{y})$ and $\boldsymbol{x}^\star$ is globally optimal. □

Another important class of optimization problems are linear programs, which have linear objective and constraint functions:

$$f_i(\alpha\boldsymbol{x} + \beta\boldsymbol{y}) = \alpha f_i(\boldsymbol{x}) + \beta f_i(\boldsymbol{y}) \tag{1.6}$$

for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$. Here, each constraint defines a halfspace. The intersection of all the halfspaces is the feasible polytope, and the optimal solution is a vertex of the polytope. Linear programming is an essential paradigm in algorithm design, forming the basis for many randomized and approximation algorithms. Figure 1.1 summarizes the relationships between the classes of optimization problems introduced so far.

1.1.2 The Five Basic Problems

A useful fact is that any convex optimization problem can be reduced to minimizing a linear function over a convex body by converting to epigraph form [8]. Grötschel, Lóvasz, and

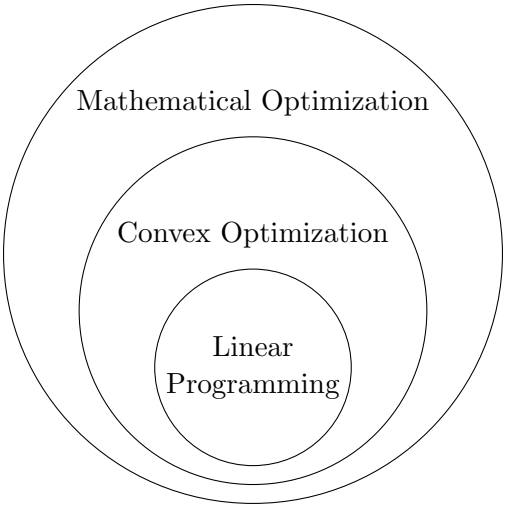


Figure 1.1: Class diagram of optimization problems.

Schrijver define five fundamental problems in this setting. Let $K \subseteq \mathbb{R}^n$ be a convex body, then for positive ϵ we define:

$$S(K, \epsilon) \coloneqq \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x} - \boldsymbol{y}\|_2 \leq \epsilon \text{ for some } \boldsymbol{y} \in K\}, \tag{1.7}$$

$$S(K, -\epsilon) \coloneqq \{\boldsymbol{x} \in K : S(\boldsymbol{x}, \epsilon) \in K\}. \tag{1.8}$$

Roughly, these two sets correspond to the points "almost inside" and "deep inside" K, respectively. The five basic problems are as follows:

1. **Weak Optimization (WOPT):** Given $\boldsymbol{c} \in \mathbb{Q}^n$ and positive $\epsilon \in \mathbb{Q}$, either assert that $S(K, -\epsilon)$ is empty, or find $\boldsymbol{y} \in \mathbb{Q}^n$ such that $\boldsymbol{y} \in S(K, \epsilon)$ and $\boldsymbol{c}^\intercal \boldsymbol{y} \leq \boldsymbol{c}^\intercal \boldsymbol{x} + \epsilon$ for all $\boldsymbol{x} \in S(K, -\epsilon)$.

2. **Weak Violation (WVIOL):** Given $\boldsymbol{c} \in \mathbb{Q}^n$, $\gamma \in \mathbb{Q}$, and positive $\epsilon \in \mathbb{Q}$, either assert that $\boldsymbol{c}^\intercal \boldsymbol{x} \geq \gamma - \epsilon$ for all $\boldsymbol{x} \in S(K, -\epsilon)$, or find $\boldsymbol{y} \in S(K, \epsilon)$ with $\boldsymbol{c}^\intercal \boldsymbol{y} < \gamma + \epsilon$.

3. **Weak Validity (WVAL):** Given $\boldsymbol{c} \in \mathbb{Q}^n$, $\gamma \in \mathbb{Q}$, and positive $\epsilon \in \mathbb{Q}$, either assert that $\boldsymbol{c}^\intercal \boldsymbol{x} \geq \gamma - \epsilon$ for all $\boldsymbol{x} \in S(K, -\epsilon)$, or assert that $\boldsymbol{c}^\intercal \boldsymbol{x} \leq \gamma + \epsilon$ for some $\boldsymbol{x} \in S(K, \epsilon)$.

4. **Weak Separation (WSEP):** Given $\boldsymbol{y} \in \mathbb{Q}^n$ and positive $\delta \in \mathbb{Q}$, either assert that $\boldsymbol{y} \in S(K, \delta)$, or find a hyperplane that almost separates $\boldsymbol{y}$ from K; that is, $\boldsymbol{c} \in \mathbb{Q}^n$

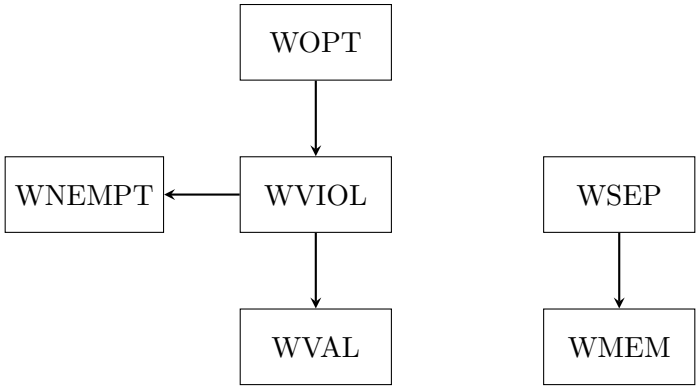


Figure 1.2: Trivial relationships between the basic problems.

with $\|\boldsymbol{c}\|_\infty = 1$ such that $\boldsymbol{c}^\intercal \boldsymbol{x} \leq \boldsymbol{c}^\intercal \boldsymbol{y} + \delta$ for every $\boldsymbol{x} \in S(K, -\delta)$.

5. **Weak Membership (WMEM):** Given $\boldsymbol{y} \in \mathbb{Q}^n$ and positive $\delta \in \mathbb{Q}$, either assert that $\boldsymbol{y} \in S(K, \delta)$, or that $\boldsymbol{y} \notin S(K, -\delta)$.

Each problem has a "strong" variant where $\epsilon, \delta = 0$, but the "weak" definitions are necessary to allow for K being an arbitrary convex body. Due to the approximate nature of these problems, the desired behavior is sometimes undefined. For example, in WMEM, if $\boldsymbol{y}$ is on the boundary of K, both conditions are satisfied and either output is acceptable.

Note that taking $\boldsymbol{c} = \boldsymbol{0}$ and $\gamma = 1$ in WVIOL reduces to checking whether $S(K, -\epsilon)$ is empty, and if not, finding a point in $S(K, \epsilon)$. Grötschel, Lóvasz, and Schrijver call this the Weak Nonemptiness problem (WNEMPT), though modern sources also refer to it as the convex feasibility problem. If K is a polytope, it is called the linear feasibility problem. Typically, K is allowed to be exponentially small, so solving the feasibility problem is like "searching for a needle in a haystack".

The relationship between these basic questions forms much of the foundational theory in this area. Figure 1.2 shows the immediate implications of the basic problems; we will fill in more of this chart as we continue through the chapter.

1.1.3 Oracles

One potential difficulty is that that the efficiency of algorithms solving these problems may vary depending on the description of K. For example, polytopes can be described as a set of explicit halfspace constraints (an H-description) or as the convex hull of a finite set of points (a V-description). Solving WSEP is trivial on an H-description but not on a V-description, while WOPT is vice versa. And though every polytope has both an H-description and a V-description, it can take exponential time to convert between them.

The solution is to represent K *implicitly*. That is, instead of explicitly describing K or its constraints, we are provided an oracle for K: an algorithm which we can query for some information about the set. In particular, the two canonical oracles are the *membership oracle*, which solves WMEM, and the *separation oracle*, which solves WSEP. These two oracles are natural to implement in many settings; membership oracles simply check feasibility, and separation oracles correspond to finding a violated constraint in the linear or convex program. Instead of analyzing runtime with respect to the explicit description of K, we say an algorithm runs in oracle-polynomial time if it has polynomial oracle complexity (measured by the worst-case number of oracle queries).

The questions explored in this thesis are centered around oracles. In particular, which oracles can solve the feasibility problem, and how powerful are they in relation to one another? We will see the importance of this question in the next section.

1.2 The Ellipsoid Method

In 1979, a new discovery shook the field of mathematical optimization. Prior to this time, it was unknown (though conjectured) that linear programming could be solved in polynomial time in the worst case, let alone general convex problems. Khachiyan resolved this major open question when he showed how the Ellipsoid Method, originally designed by Shor, Yudin, and Nemirovskii for nonlinear optimization, could be adapted to solve the linear feasibility problem in polynomial time [32, 37, 23]. As we will see, this triumph

implied efficient algorithms for linear programming and approximate convex optimization and solidified nontrivial implications between the five basic problems.

1.2.1 The Basic Algorithm

Here, we will briefly and informally describe the Ellipsoid Method for solving the convex feasibility problem for a convex body $K \subseteq \mathbb{R}^n$. We require four inputs:

1. An efficient separation oracle for K.
2. An ellipsoid $E(\boldsymbol{c}, \mathbf{Q})$ containing K, with $\boldsymbol{c} \in \mathbb{R}^n$ the center and $\mathbf{Q}$ the positive semi-definite matrix describing the ellipsoid.
3. Positive $R \in \mathbb{Q}$ satisfying $\mathbf{vol}(E) \leq R$.
4. Positive $r \in \mathbb{Q}$ such that if K is nonempty, then $\mathbf{vol}(K) \geq r$.

The Ellipsoid Method runs as follows: we begin with E and use the separation oracle to query whether $\boldsymbol{c} \in K$. If so, we terminate and output $\boldsymbol{c}$. Otherwise, we obtain a separating hyperplane $\boldsymbol{h}$ such that K is contained in the half-ellipsoid $E \cap \{\boldsymbol{y} : \boldsymbol{h}^\intercal \boldsymbol{y} \leq \boldsymbol{h}^\intercal \boldsymbol{c}\}$. Then, we let $E' = E(\boldsymbol{c}', \mathbf{Q}')$ be the minimum volume ellipsoid containing the half ellipsoid. If $\mathbf{vol}(E') \geq r$ then we repeat the separating hyperplane query with E'; otherwise we return that K is empty. Notice that we did not need r here, but it made things simpler; because we are solving the weak problem, we can just run the Ellipsoid Method until $\mathbf{vol}(E) < \epsilon$.

The crucial part of the algorithm is calculating $\boldsymbol{c}'$ and $\mathbf{Q}'$ to obtain the minimum volume ellipsoid. This is called the Löwner-John ellipsoid, and it is hard to compute in general, but in this case there exists an explicit formula. Additionally, if K is not full-dimensional, we require some additional techniques related to simultaneous Diophantine approximation.

A useful lemma, with a concise proof available in [30], is that

$$\frac{\mathbf{vol}(E')}{\mathbf{vol}(E)} \leq e^{-\frac{1}{2(n+1)}}. \tag{1.9}$$

Therefore, after $2(n+1)\ln\frac{R}{r}$ iterations, we have $\mathbf{vol}(E) \leq r$, triggering termination. This also implies that we can solve the feasibility problem in polynomial time when the volume

ratio $\frac{R}{r}$ is exponential; that is, the Ellipsoid Method enables us to solve the "needle in a haystack" problem given access to an efficient separation oracle.

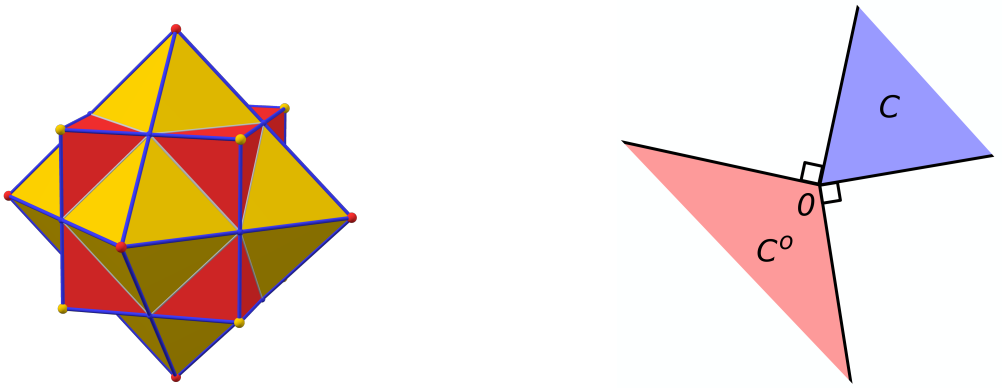
1.2.2 Solvability of Convex Optimization

The Ellipsoid Method allows us to show that WSEP implies WOPT. We show this by way of WVIOL, because once we have a polynomial-time algorithm for WVIOL, we can binary search over γ to solve WOPT. Assume we are given $\boldsymbol{c} \in \mathbb{Q}^n$, an efficient separation oracle for the feasible region $\mathcal{X}$, and a radius R such that the ellipsoid E of radius R about the origin contains $\mathcal{X}$. Then solving WVIOL reduces to the following convex feasibility problem:

$$\begin{aligned} \text{find} \quad & \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{x} \in \mathcal{X} \\ & \boldsymbol{c}^\intercal \boldsymbol{x} < \gamma + \epsilon. \end{aligned} \tag{1.10}$$

Suppose $K \subseteq \mathbb{R}^n$ is the feasible region for this problem. We first need a separation oracle for K, which we can obtain by using the separation oracle for $\mathcal{X}$ and computing $\boldsymbol{c}^\intercal \boldsymbol{x}$. We also need an ellipsoid containing K; since $K \subseteq \mathcal{X}$ we can simply use E. Finally, we need that K is no more than exponentially smaller than E (in n and $\log(\frac{1}{\epsilon})$) for efficiency of the Ellipsoid Method. While not obvious, this is true if $\gamma \geq \boldsymbol{c}^\intercal \boldsymbol{x}^\star$, which is enough (see [13]).

This strategy is also applicable to linear programming problems. Since the vertices of the feasible polytope have polynomial bit complexity, we can "round" an ϵ-optimal solution to an optimal one. Modulo a few more numerical technicalities described in [18], we have an efficient algorithm for solving linear programming. This enables us to solve linear programs with exponentially many constraints in polynomial time – that is, without even looking at the entire program.



(a) A cube (red) and its polar (yellow).

(b) A cone C and its polar C°.

Figure 1.3: Geometric examples of polarity. Images from [2, 31].

1.3 Equivalence of Separation and Optimization

In the previous section, we described how the breakthrough Ellipsoid Method showed WSEP $\overset{R}{\Longrightarrow}$ WOPT. It is a surprising and deep result that the converse is true, and we do not even need R. To show this, we need the concept of geometric duality, or polarity. The standard way to describe a convex body $S \subseteq \mathbb{R}^n$ is just the set of points comprising it: $S = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} \in S\}$. The dual description is that S is the intersection of all closed halfspaces $\mathcal{H}$ containing it: $S = \bigcap_{H \in \mathcal{H}} H$.

Suppose S contains the origin, then one way to represent $\mathcal{H}$ is via another convex body S°, called the polar of S. It is defined as follows:

$$S^\circ = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y}^\intercal \boldsymbol{x} \leq 1 \ \ \forall \boldsymbol{x} \in S\}. \tag{1.11}$$

S° can be thought of as the normalized representation of halfspaces containing S. In addition, S° contains the origin and $S^{\circ\circ} = S$. For polytopes it is especially elegant: if P is a polytope with constraint matrix $\mathbf{A}$ and vector $\mathbf{1}$, then P° is the convex hull of the rows of $\mathbf{A}$. Thus, facets of P correspond to vertices of P° and vice-versa, as in Figure 1.3a.

Another important polar description is that of cones. Recall that a convex cone C contains $\mathbf{0}$ and the ray from $\mathbf{0}$ through any point in C. If C is a polyhedral cone, then it is

the conic hull of finitely many fundamental vectors. We have

$$C^\circ = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y}^\intercal \boldsymbol{x} \leq 0 \ \ \forall \boldsymbol{x} \in C\}, \tag{1.12}$$

since for cones, $\boldsymbol{y}^\intercal \boldsymbol{x} \leq 0 \iff \boldsymbol{y}^\intercal \boldsymbol{x} \leq 1$. See Figure 1.3b for a geometric interpretation.

Polarity allows us to connect WOPT and WSEP; in particular, separation over S reduces in constant time to optimization over S°, and vice-versa since $S^{\circ\circ} = S$. A complete proof is available in [18], but for this illustration from [13] assume we have access to a strong separation oracle (*i.e.,* a separation oracle solving WSEP with $\delta = 0$).

Lemma 1. *Strong separation over a convex body $S \subseteq \mathbb{R}^n$ is equivalent to strong optimization over its polar S°.*

Proof. Suppose we wish to solve the separation problem for $\boldsymbol{x} \in \mathbb{R}^n$. By polarity, $\boldsymbol{x} \in S$ if and only if $\boldsymbol{y}^\intercal \boldsymbol{x} \leq 1$ for all $\boldsymbol{y} \in S^\circ$. Let

$$m = \boldsymbol{y}^{\star\intercal} \boldsymbol{x} = \max_{\boldsymbol{y} \in S^\circ} \boldsymbol{y}^\intercal \boldsymbol{x}, \tag{1.13}$$

then $\boldsymbol{y}^\intercal \boldsymbol{x} \leq 1$ for all $\boldsymbol{y} \in S^\circ$ if and only if $m \leq 1$. We can compute m by solving the optimization problem for S°. If $m \leq 1$ then $\boldsymbol{x} \in S$; otherwise $\boldsymbol{y}^\star$ is a separating hyperplane. □

1.4 The Yudin-Nemirovskii Theorem

With the separation problem well-understood, we now turn attention to the membership problem. The most important result in this area is the Yudin-Nemirovskii Theorem, critical because it quantifies a fundamental difference in solving power between separation oracles and membership oracles. The theorem states that WMEM implies WOPT, but unlike WSEP, we additionally need an initial point $\boldsymbol{a_0}$ inside the convex body $K \subseteq \mathbb{R}^n$, as well as positive $r \in \mathbb{Q}$ such that if K is nonempty, then $\mathbf{vol}(K) \geq r$. This is because, unlike separation oracles, membership oracles are not strong enough to solve the feasibility problem (they would need to query exponentially many points).

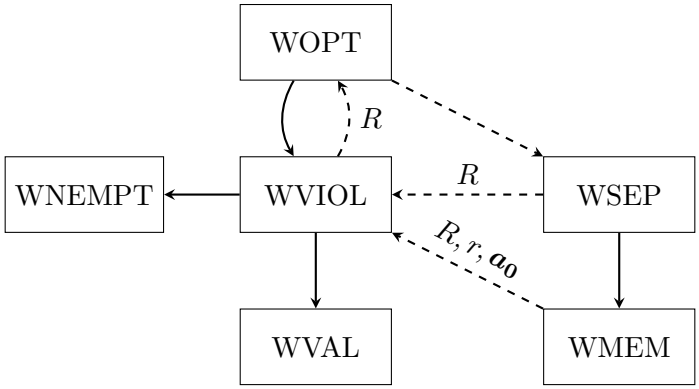


Figure 1.4: Major relationships between the basic problems. Solid lines indicate trivial relationships, while dashed lines indicate results implied by the Ellipsoid Method, which efficiently solves WNEMPT.

The most natural way to show WMEM implies WOPT is by using the membership oracle to derive a separation oracle and invoke the reduction from Section 1.3. However, this strategy proved elusive and was not invented until 2017 (see Section 2.3.1); the result required a recent lemma regarding the approximation of Lipschitz convex functions and implemented a sophisticated randomized separation oracle.

Yudin and Nemirovskii's derivation [18] instead uses the membership oracle to solve the violation problem directly. First, they show that any membership oracle can be used to strengthen itself; that is, it results in an oracle which given $\boldsymbol{y} \in \mathbb{Q}^n$ and positive $\delta \in \mathbb{Q}$, either asserts that $\boldsymbol{y} \in S(K, \delta)$ or that $\boldsymbol{y} \notin K$. This oracle enables the derivation of a "very weak" separation oracle, which is not quite as fast or accurate as a true separation oracle. However, it is sufficient to implement an advanced version of the Ellipsoid Method called the shallow-cut Ellipsoid Method, which can efficiently solve the violation problem.

Combining our results from the previous sections, we obtain the implications chart shown in Figure 1.4. For a more complete treatment of the basic problems, including the fact that the assumptions on R, r, and $\boldsymbol{a_0}$ cannot be weakened, see Chapter 4 of Grötschel, Lóvasz, and Schrijver's book [18].

Chapter 2

A Survey of Oracle Methods for Convex Optimization

2.1 Introduction

The Ellipsoid Method for solving the convex feasibility problem, and the resultant equivalence of separation and optimization, enables a unique situation where any of the basic problems can be solved in polynomial time given an efficient separation oracle. It is natural to ask whether faster algorithms exist and what happens when we cannot obtain a separation oracle. Surprisingly, this line of research has unearthed interesting interdisciplinary questions in complexity, learning theory, approximation algorithms, and mechanism design, with applications to nearly all real-world areas where convex optimization is utilized.

In this chapter, we present a brief survey of recent work in oracle methods for solving the feasibility problem and summarize its major directions. While the majority of work in this area has focused on improving the oracle complexity and runtime of the Ellipsoid Method, separation oracle-based algorithms are not the only option. In many cases, the separation problem is $\mathcal{NP}$-hard, preventing the implementation of an efficient separation oracle; several notions of approximation have been developed to address this. In addition, it is interesting to study the theoretical power and limitations of certain oracles. As the simplest type of

oracle, the membership oracle has received much attention. Likewise, the index oracle is important for gauging how little information is required to solve linear programs. And on the frontier, quantum membership oracles have unlocked even faster algorithms for the convex feasibility problem.

2.2 Cutting-Plane Methods

The Ellipsoid Method is the first in a class of algorithms which solve the feasibility problem for a convex body $K \subseteq \mathbb{R}^n$ called *cutting-plane methods*. In general, cutting-plane methods iteratively refine a feasible region $\Omega \subseteq \mathbb{R}^n$ via the separation oracle [21]. In each iteration, the separation oracle is queried at $\boldsymbol{x} \in \Omega$; if $\boldsymbol{x} \in K$, we are done. Else, we use the halfspace H returned by the separation oracle to reduce the volume of Ω and compute a new candidate point. The Ellipsoid Method maintains Ω an ellipsoid and $\boldsymbol{x}$ its center, then updates Ω to be the smallest ellipsoid containing $\Omega \cap H$.

The story of the development of faster cutting-plane methods is one of iteratively improving oracle complexity (the amount of calls to the separation oracle) and runtime complexity (roughly the number of operations of the algorithm). Each value depends on the dimension n and the ratio $\kappa = \frac{nR}{\epsilon}$, which encodes the extra difficulty of solving the feasibility problem in higher dimension, with a larger bounding ellipsoid, or at a more precise level of approximation. The oracle complexity of cutting-plane methods is $\Omega(n \log(\kappa))$ [29] and the runtime complexity is conjectured $\Omega(nT \log(\kappa) + n^3 \log(\kappa))$, where T is the runtime of the separation oracle [21]. It is worth noting that T can be large, causing the oracle complexity term to dominate the overall runtime; for example, a separation oracle for the cone of positive semi-definite matrices must run an eigenvector computation, which can take time $O(n^3)$ [35, 12]. Refer to Table 2.1 for a list of algorithms and their complexities.

The Ellipsoid Method is suboptimal in both categories, with a particularly poor oracle complexity of $O(n^2 \log(\kappa))$ since it discards halfspace computations from previous iterations. However, one property of the Ellipsoid Method is that its per-iteration runtime is only

Reference	Year	Method	Oracle Complexity	Runtime Complexity
[27]	1965	Center of Gravity	$n\log(\kappa)$	Exponential
[32, 37, 23]	1979	Ellipsoid	$n^2\log(\kappa)$	$n^2T\log(\kappa)+n^4\log(\kappa)$
[24]	1988	Inscribed Ellipsoid	$n\log(\kappa)$	$nT\log(\kappa)+(n\log(\kappa))^{4.5}$
[33]	1989	Volumetric Center	$n\log(\kappa)$	$nT\log(\kappa)+n^{\omega+1}\log(\kappa)$
[4]	1995	Analytic Center	$n\log^2(\kappa)$	$nT\log^2(\kappa)+n^{\omega+1}\log^2(\kappa)+(n\log(\kappa))^{2+\omega/2}$
[6]	2004	Random Walk	$n\log(\kappa)$	$nT\log(\kappa)+n^7\log(\kappa)$
[26]	2015	Hybrid Center	$n\log(\kappa)$	$nT\log(\kappa)+n^3\log^3(\kappa)$
[21]	2020	Volumetric Center	$n\log(\kappa)$	$nT\log(\kappa)+n^3\log(\kappa)$

Table 2.1: Cutting-plane methods for the convex feasibility problem [21]. The final algorithm is optimal in oracle and (conjectured) runtime complexity. Here, $\kappa=(nR)/\epsilon$, T is the runtime of the separation oracle, and $\omega<2.373$ is the matrix multiplication constant.

$O(n^2)$, faster than all future methods, due to the simplicity of the ellipsoid update [21].

A decade after the Ellipsoid Method, two algorithms were published which achieved optimal oracle complexity. They both achieve this bound by reusing the information from previous oracle calls to avoid excessive queries. Khachiyan *et al.* [24] developed the Method of Inscribed Ellipsoids, which takes Ω the intersection of all halfspaces given by the separation oracle (forming a polytope), and $\boldsymbol{x}$ the center of the Löwner-John ellipsoid of Ω (that is, the maximum-volume ellipsoid inscribable in Ω). This method took inspiration from the (much earlier) Center of Gravity Method due to Levin [27], which takes $\boldsymbol{x}$ the center of gravity of Ω; however, finding the center of gravity of a polytope is #P-hard, while the Löwner-John ellipsoid can be computed in polynomial time [19].

The Volumetric Center Method [33], also called Vaidya's Method, sped this up by maintaining an approximation of Ω and taking $\boldsymbol{x}$ its volumetric center: a point through which a hyperplane has a good chance of dividing Ω into two parts of approximately equal volume. Vaidya also relied on *leverage scores*, which encoded the relative importance of each hyperplane for the approximation of Ω. However, this method was still not very efficient because updating leverage scores each iteration involved a matrix inversion computation (hence the matrix multiplication constant ω in the runtime).

In the next two decades, several algorithms were developed that did not improve the overall runtime, but still introduced novel and useful techniques. The Analytic Center

Method [4] developed by Atkinson and Vaidya uses a different type of center which avoids matrix inversion, but is ultimately less efficient. One advantage of the this method, however, is that the leverage scores need not be as precise as in Vaidya's Method. Approximating leverage scores in this way remained a theme in the literature into the 2010s. The Random Walk Method, developed by Bertsimas and Vempala [6], finds the approximate center of gravity of Ω by performing a random walk in the polytope. This is a fairly expensive computation, but enables solving a slight generalization of the feasibility problem corresponding to minimizing a quasi-convex function.

In 2015, Lee *et al.* [26] developed the first algorithm to improve runtime on the feasibility problem since Vaidya. Their method improved on Vaidya's Method by using the random Johnson-Lindenstrauss projection to approximate changes in leverage scores. They also took advantage of the fact that leverage scores only change slightly between iterations to enable computation in amortized $\tilde{O}(n^2)$ time. This is called the Hybrid Center method because the function minimized to find the center is a weighted combination of the functions corresponding to the volumetric and analytic centers.

Five years later, Jiang joined Lee *et al.* [21] to develop an algorithm which matches the conjectured lower bound on the runtime of cutting-plane methods. Their approach does away with the hybrid center to remove the extra $log^2(\kappa)$ term in the runtime and runs Vaidya's Method in "phases" to control error accumulation. They also use some recent advances from numerical linear algebra such as fast rectangular matrix multiplication to design an efficient data structure for leverage score maintenance. Interestingly, if $\omega = 2$ as conjectured [11], then Vaidya's Method is optimal (and much simpler).

2.3 Convex Optimization Without a Separation Oracle

Besides their application in cutting-plane methods, separation oracles are well-studied because they have a natural formulation in many algorithm design problems: oftentimes, finding a separating hyperplane simply corresponds to finding a violated constraint in the

Reference	Year	Method	Oracle Complexity
[18]	1988	Violation Oracle	$O(n^{10})$
[6]	2004	Random Walk	$\tilde{O}(n^5)$
[22]	2006	Simulated Annealing	$\tilde{O}(n^{4.5})$
[1]	2016	Simulated Annealing	$\tilde{O}(\sqrt{\nu}n^4)$
[25]	2017	Separation Oracle	$\tilde{O}(n^2)$

Table 2.2: Methods for convex optimization with a membership oracle. $\tilde{O}$ hides logarithmic factors, *e.g.*, $\tilde{O}(n^2) = O(n^2 \log^{O(1)}(\frac{nR}{\epsilon r}))$. ν is a parameter dependent upon the structure of the set, *e.g.*, $\nu = O(\sqrt{n})$ for the cone of semi-definite matrices.

linear or convex program. As seen in Section 2.2, algorithms have been developed which match the lower bound on oracle complexity and the conjectured lower bound on runtime complexity in this setting. But not every problem easily admits a separation oracle, and analyzing the feasibility problem in different oracle settings can result in theoretical insights and faster algorithms. As a result, an important research initiative is to understand alternative oracles for solving the feasibility and optimization problems.

2.3.1 Membership Oracles

In the absence of a separation oracle, researchers first turned to the weaker oracle specified by Grötschel, Lóvasz, and Schrijver: the membership oracle [18]. In Section 1.4, we showed that the Yudin-Nemirovskii Theorem implies that a membership oracle cannot solve the feasibility problem; thus, they are only able to solve weak optimization when given a point $\boldsymbol{a_0} \in K$ to begin with. Despite this, there has been much interesting work on improving the runtime of the WMEM $\Longrightarrow$ WOPT reduction. This line of work is theoretically motivated because it advances understanding of the power and limitations of the most basic oracle, and also important for applications because membership oracles are often trivial or easy to obtain in a variety of problem settings. Refer to Table 2.2 for a summary of major algorithms and their oracle complexities.

The system of Grötschel, Lóvasz, and Schrijver [18] to solve WOPT using a membership oracle involves a sophisticated variant of the Ellipsoid Method called the shallow-cut

Ellipsoid Method and has an enormous oracle complexity of $O(n^{10})$ [25]. The reason for this is fairly simple. In 1988, there was no known way to directly, efficiently implement a separation oracle with a membership oracle. To accomplish this, one had to implement a violation oracle using a membership oracle and the Ellipsoid Method, then utilize a polarity argument and the Ellipsoid Method again to solve the separation problem.

There was little significant progress in this area until 2004 with the publication of the Random Walk Method [6] described in Section 2.2. This method uses the membership oracle to perform a random walk on the set and compute an approximate center of gravity; it is an extension of the same algorithm for a separation oracle, but pays for the weaker membership oracle with an increased oracle complexity of $\tilde{O}(n^5)$.

The next improvement came in the form of the simulated annealing technique, a random walk-based method for optimization which searches the solution space with respect to a variable called "temperature", which slowly biases in favor of optimal solutions. Kalai and Vempala's algorithm [22] based on this technique improved the bound to $\tilde{O}(n^{4.5})$. A decade later, Abernethy and Hazan [1] developed a better temperature schedule to further improve the runtime to $\tilde{O}(\sqrt{\nu}n^4)$, where ν is a parameter dependent upon the structure of the set. For example, $\nu = O(\sqrt{n})$ for the cone of semi-definite matrices, an improvement of $O(\sqrt{n})$ over the original simulated annealing algorithm.

Finally, in 2017, Lee, Sidford, and Vempala [25] solved the original problem in Grötschel, Lóvasz, and Schrijver's reduction by showing how to directly implement a separation oracle with a membership oracle. They reduce the separation problem to computing an approximate subgradient of a Lipschitz convex function, which can be solved by an evaluation oracle. Interestingly, the resultant separation oracle is randomized (*i.e.,* it has nonzero failure probability) and must be run multiple times to guarantee success with high probability; it is doubtful that obtaining a deterministic separation oracle is possible in this setting. Their method has an excellent oracle complexity of $\tilde{O}(n^2)$ and has resulted in more efficient reductions between Grötschel, Lóvasz, and Schrijver's five basic problems (as defined in Section 1.1.2).

2.3.2 Alternative Oracles

Besides membership oracles, which have been well-studied since Grötschel, Lóvasz, and Schrijver [18], much recent research has focused on developing novel oracles for specific optimization situations. Some particularly interesting alternative oracles include approximate separation oracles and index oracles for their combination of practical applicability and theoretical insight. The most recent invention, quantum oracles, have resulted in faster algorithms for solving the feasibility problem.

Approximate Oracles

In 2002, Jansen [20] developed a notion of approximate separation oracles for polytopes. Since optimization and separation are equivalent (see Section 1.3), if solving the optimization problem (or rather, its decision variant the violation problem) is $\mathcal{NP}$-hard, the separation problem is also $\mathcal{NP}$-hard. Jansen noticed that this was the case in problems such as such as fractional graph coloring and preemptive scheduling, and he developed an approximation algorithm in the following setting. Suppose $K \subseteq \mathbb{R}^n$ is a fractional packing polytope, that is the intersection of two polytopes $A \cap B$ where the separation problem is $\mathcal{NP}$-hard for A. Denote by K^α the polytope formed by scaling A by a factor $\alpha \geq 1$ and intersecting it with B. Then we have:

1. **Weak Approximate Optimization (WAOPT):** Given $\boldsymbol{c} \in \mathbb{Q}^n$ and positive $\epsilon \in \mathbb{Q}$, either assert that $S(K, -\epsilon)$ is empty, or find $\boldsymbol{y} \in \mathbb{Q}^n$ such that $\boldsymbol{y} \in S(K, \epsilon)$ and $\boldsymbol{c}^\intercal \boldsymbol{y} \leq \alpha \boldsymbol{c}^\intercal \boldsymbol{x} + \epsilon$ for all $\boldsymbol{x} \in S(K, -\epsilon)$.

2. **Weak Approximate Separation (WASEP):** Given $\boldsymbol{y} \in \mathbb{Q}^n$ and positive $\delta \in \mathbb{Q}$, either assert that $\boldsymbol{y} \in S(K^\alpha, \delta)$, or find a hyperplane that almost separates $\boldsymbol{y}$ from K; that is, $\boldsymbol{c} \in \mathbb{Q}^n$ with $\|\boldsymbol{c}\|_\infty = 1$ such that $\boldsymbol{c}^\intercal \boldsymbol{x} \leq \boldsymbol{c}^\intercal \boldsymbol{y} + \delta$ for every $\boldsymbol{x} \in S(K, -\delta)$.

Notice that if $\alpha = 1$, we recover the original problems of Grötschel, Lóvasz, and Schrijver [18] (see Section 1.1.2). A weak approximate separation oracle solves WASEP. Jansen also developed a modified Ellipsoid Method using this oracle to solve the weak approximate

nonemptiness problem, and thereby WAOPT. This resulted in a number of new approximation algorithms and polynomial-time approximation schemes (PTAS) for $\mathcal{NP}$-hard problems described by a fractional packing polytope.

Various other notions of approximate separation oracles have been studied in the literature, oftentimes tuned to problem specifications. In the context of Bayesian persuasion, Dughmi and Xu [16] developed the one-sided-error separation oracle (OSO) which generalizes the notion of oracle approximation. In their case, oracle error is quantified in terms of earthmover distance, which does not correspond to a typical p-norm distance; as a result, the oracle may accept points that are ϵ-optimal with respect to earthmover distance but far from feasible with respect to the 2-norm distance. They analyze a modified version of the Ellipsoid Method which is ϵ-optimal with respect to the points the OSO accepts.

To accommodate approximation algorithms and bi-criterion optimization, Weinberg [34] developed the weird separation oracle (WSO), with "weird" signifying its erratic behavior. A WSO is defined with respect to a multiplicatively scaled convex set in the sense of Jansen [20], but may accept points in a "weird" region that may not be convex, closed, or even connected. Weinberg proves that using the WSO we can find ϵ-optimal points with respect to the accepted region; he also uses a WSO version of the Ellipsoid Method to generalize the equivalence of separation and optimization (see Section 1.3) to bi-criterion optimization.

Index Oracles

Bei *et al.*'s [5] work, which directly inspired this thesis, studied the following question: What is the least amount of information, in what format, needed to solve linear programming efficiently? To do so, they designed a new type of oracle which they call a verification oracle, but I term an index oracle. For a polytope $K = \{\boldsymbol{x} \in \mathbb{R}^n : \mathbf{A}\boldsymbol{x} \leq \boldsymbol{b}\}$, they compare the relative strengths of two index oracles:

1. **Furthest Oracle:** An algorithm which, given $\boldsymbol{y} \in \mathbb{R}^n$, decides whether $\boldsymbol{y} \in K$, and

if not, finds the index of the most violated constraint; that is, the index i such that

$$i = \arg\max \left\{ \frac{b_i - \boldsymbol{a_i}^\intercal \boldsymbol{y}}{\|\boldsymbol{a_i}\|} : \boldsymbol{a_i}^\intercal \boldsymbol{y} > b_i \right\}. \tag{2.1}$$

2. **Worst-Case Oracle:** An algorithm which, given $\boldsymbol{y} \in \mathbb{R}^n$, decides whether $\boldsymbol{y} \in K$, and if not, finds the index of a violated constraint; that is, an (adversarially chosen) index i such that $\boldsymbol{a_i}^\intercal \boldsymbol{y} > b_i$.

Bei *et al.* give application scenarios where index oracles are a natural choice, such as signal interference, and they surprisingly show that furthest oracles can solve the (strong) feasibility problem efficiently while the worst-case oracle cannot. In particular, the furthest oracle can solve the feasibility problem in oracle complexity polynomial in the dimension n, the number of constraints m, and the bit complexity L of the polytope, while the worst-case oracle requires queries exponential in n. In fact, this setting has a lower bound of $\Omega(m^{\lfloor n/2 \rfloor})$ queries for any randomized algorithm.

To obtain the result for the furthest oracle, Bei *et al.* use a clever geometric argument. They view $\mathbf{A}$ as a point in $\mathbb{R}^{mn}$; a point is a degenerate polytope, so they use the Ellipsoid Method to find $\mathbf{A}$. Suppose $\mathbf{A'}$ is the center of the ellipsoid. They first project the rows of $\mathbf{A'}$ onto the sphere S^{n-1} and consider the resultant nearest-neighbor partition, called the Voronoi diagram. They show that, since the furthest oracle considers the most violated constraint, the index returned by the oracle for an input $\boldsymbol{x} \in \mathbb{R}^n$ is exactly the furthest Voronoi cell that contains $\boldsymbol{x}$. They use this fact to check whether the Voronoi diagram of $\mathbf{A}$ is equivalent to the Voronoi diagram of $\mathbf{A'}$; if so, we recover $\mathbf{A}$, and solve the feasibility problem, and if not, we obtain the necessary separating hyperplane between $\mathbf{A}$ and $\mathbf{A'}$.

Notice that m is a factor in the oracle complexity using the furthest oracle not present in that of the Ellipsoid Method. Using IND-F is much slower than the Ellipsoid Method as m becomes large, and it is impossible to solve feasibility for a convex set since $m \to \infty$. However, the point of Bei *et al.*'s work was to show that solving linear feasibility efficiently with an index oracle is *possible*, rather than practical.

Quantum Oracles

In late 2019, two independent studies released a day apart [3, 10] developed a theory of quantum membership oracles for solving convex optimization. In contrast to the state-of-the-art algorithm [25] for optimization with a membership oracle, which has oracle complexity $\tilde{O}(n^2)$, the quantum algorithm only makes $\tilde{O}(n)$ queries. The main advantage of a quantum membership oracle is that it can query a superposition of vectors simultaneously. This enables the speedup of the computation of approximate subgradients of convex Lipschitz functions, the technique of [25] for recovering a separation oracle. While the results of the two works are similiar, their strategies differ greatly: [3] applies Jordan's quantum algorithm for gradient computation, while [10] combines classical randomness and mollifier functions. In addition, both works discover lower bounds on quantum oracle complexity: the optimization problem is $\Omega(\sqrt{n})$ when we have a feasible point, and $\Omega(n)$ when we need to solve the feasibility problem. However, they conjecture that the first case is $\tilde{\Theta}(n)$, meaning that possession of a feasible point has little benefit for query complexity. Furthermore, the runtime complexity of the quantum algorithm is $\tilde{O}(n^3)$, the same as the classical algorithm; it is conjectured that a quantum reduction from optimization to separation would improve this runtime.

Chapter 3

The Distance Oracle and the Oracle Power Hierarchy

3.1 Introduction

This chapter is composed of original work. We propose a novel type of oracle for solving the feasibility problem called the distance oracle, which returns the minimum Euclidean distance from the query point to the feasible region. We show progress on solving the linear feasibility problem with the distance oracle in $O(n \log(n))$ queries, a logarithmic improvement over the best separation oracle-based method. We develop a framework for analyzing oracle power and show that distance is stronger than separation. Finally, we describe some open questions in this area, whose answers would increase the comprehensiveness of our work.

Studying the distance oracle is important for both theory and application. First, while a separation oracle is critical for cutting-plane methods (see Section 2.2), it is not always easy to obtain, for reasons ranging from $\mathcal{NP}$-hardness of the separation problem to the real-world constraints of certain applications. Second, while many alternative oracles have been proposed (see Section 2.3), only quantum oracles have resulted in optimization speedups. In stark contrast to quantum approaches, separation oracles still reign as the most "natural" oracle, as deriving one often involves simply finding a violated constraint.

To this end, the distance oracle is a natural, applicable classical oracle which takes advantage of the geometry of the feasibility problem to enable efficient algorithms.

For contributions to this chapter, I would especially like to thank Shaddin Dughmi for suggesting the distance oracle and guidance on the proofs, and Max Daniels for fruitful conversations on the distance oracle and suggesting the triangulation method.

An Application

The distance oracle lends itself to problems which are intrinsically geometric in nature, where there is a natural notion of distance but the interpretation of a separating hyperplane is often unclear. One interesting example is a variant of the interactive learning problem. In the model of [17], there is an optimal model s^* we are trying to learn. The learning algorithm may propose a model s and receive user feedback in the form of model s' dependent on s such that s' is "more similar" to s^* than s was. If $s = s^*$, the feedback model is simply s. In [17] they consider a robust model where feedback may be misleading, but here we assume the user feedback is always correct.

At first sight, this problem seems like it may lend itself to a distance oracle, since it is geometric in nature and the user feedback requires some notion of distance. In particular, suppose we are trying to learn a probabilistic classifier s^* in $[0, 1]^n$. Then, our proposal region is a box of side length $R = 1$. The distance oracle, encoding the user feedback which measures the "similarity" of a proposed classifier to the true classifier, would return the Euclidean distance of a proposed point $\boldsymbol{y} \in \mathbb{R}^n$ to s^*. In the next section, we will show that this is enough information to recover s^* in $O(n \log(n))$ oracle queries.

While the setting as described above is slightly unrealistic, some extensions to our work can improve this. In particular, extending our work to the weak convex feasibility problem would enable us to find an approximately optimal classifier when the set of "acceptable" classifiers is convex. For measuring distance between classifiers, statistical distance may be more realistic than Euclidean distance; applying the distance oracle to other metrics is an important question for future work. Also, as in [17], the user feedback may not always be

correct, so studying the robustness of the distance oracle is another interesting direction.

3.2 The Distance Oracle

Suppose $K \subseteq \mathbb{R}^n$ is a full-dimensional convex body. We define the Minimum Distance problem (DIS) as follows: Given $\boldsymbol{y} \in \mathbb{R}^n$, either assert that $\boldsymbol{y} \in K$ or find the minimum Euclidean distance $D(\boldsymbol{y})$ from $\boldsymbol{y}$ to K; that is, $D(\boldsymbol{y}) = \min_{\boldsymbol{x} \in K} \|\boldsymbol{y} - \boldsymbol{x}\|_2$. Let $C(\boldsymbol{y})$ be the minimizer of this function (*i.e.,* $C(\boldsymbol{y})$ is the closest point in K to $\boldsymbol{y}$). A distance oracle solves DIS.

Clearly the distance oracle correctly answers membership queries, so by [25] it can solve convex optimization in $\tilde{O}(n^2)$ queries given a point $\boldsymbol{a}_0 \in K$ (see Section 2.3.1). If we are not given $\boldsymbol{a}_0$, we must solve the feasibility problem. We will show progress on a result for the linear version that requires $O(n \log(n))$ queries.

A simple lemma is that for any $\boldsymbol{y} \notin K$, we have $C(\boldsymbol{y}) \in \partial(K)$; that is, $C(\boldsymbol{y})$ is on the boundary of K. In the linear case, K is a polytope, so it seems natural to consider separately the cases where $C(\boldsymbol{y})$ is on a facet ($n-1$ dimensional face), ridge ($n-2$ dimensional face), vertex (0-dimensional face), and so on.

3.2.1 Finding a Facet

In the following algorithm, we use the fact that facets of K are hyperplanes to find a point in K by querying a basis around $\boldsymbol{y}$. See Figure 3.1 for a visualization.

Proposition 1. *Suppose H is a facet of K with unit normal vector $\boldsymbol{h}$, and $C(\boldsymbol{y}) \in H$ strictly for a query point $\boldsymbol{y} \in \mathbb{R}^n$. Then we can recover $C(\boldsymbol{y})$ in $O(n)$ distance oracle queries, solving the feasibility problem.*

Proof. Suppose $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ is a basis for $\mathbb{R}^n$ where $\|\boldsymbol{b}_i\|_2 = \epsilon$ for all $i \in \{1, \ldots, n\}$. We need ϵ small enough such that $C(\boldsymbol{y} + \boldsymbol{b}_i) \in H$ for all i. We will specify this quantity later.

Let $d = D(\boldsymbol{y})$. For each $i \in \{1, \ldots, n\}$ we set $d_i = D(\boldsymbol{y} + \boldsymbol{b}_i)$. This requires $n + 1$ calls

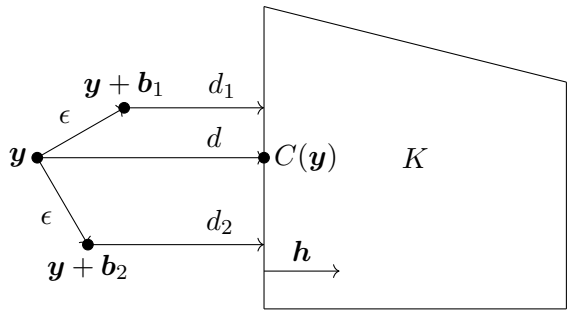


Figure 3.1: Querying an ϵ-basis around $\boldsymbol{y}$ to find a point in K.

to the distance oracle. By the definition of dot product,

$$d_i = d - \boldsymbol{b}_i{}^{\mathsf{T}}\boldsymbol{h}. \tag{3.1}$$

Let

$$\mathbf{B} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \text{ and } \boldsymbol{d} = \begin{bmatrix} d - d_1 \\ d - d_2 \\ \vdots \\ d - d_n \end{bmatrix}. \tag{3.2}$$

Since all the $\boldsymbol{b}_i$ are linearly independent, $\mathbf{B}$ is nonsingular. Thus, we have the system of linear equations

$$\mathbf{B}\boldsymbol{h} = \boldsymbol{d}, \tag{3.3}$$

and by solving this system, we obtain the unique solution $\boldsymbol{h}$. Then,

$$\boldsymbol{y} + d\boldsymbol{h} = C(\boldsymbol{y}) \in K. \tag{3.4}$$

Notice this computation is coordinate-free, and therefore the algorithm is rotation-invariant.

□

Notice that Proposition 1 does not work if H is not a facet because there will be no ϵ such that $C(\boldsymbol{y}+\boldsymbol{b}_i) \in H$ for all i. While this algorithm may seem promising, it is ultimately ineffective because almost the entire space is closest to some vertex of K rather than a facet; we will show this in the next section (see Fact 2).

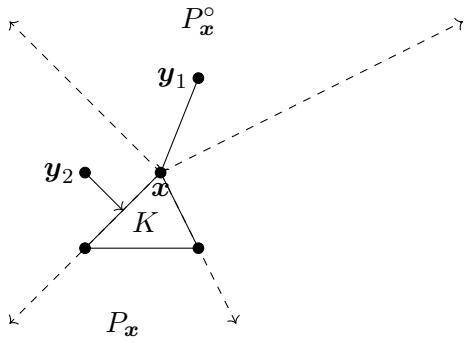


Figure 3.2: A point $\boldsymbol{y} \in \mathbb{R}^n$ is closest to a vertex $\boldsymbol{x}$ of K if and only if $\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ$. Here, $\boldsymbol{y}_1$ is closest to the vertex $\boldsymbol{x}$, but $\boldsymbol{y}_2$ is closest to a facet of K.

3.2.2 Finding a Vertex

We will next consider how to find a point in K if the query point $\boldsymbol{y}$ is closest to a vertex of K. First, we geometrically characterize these points. Let $\boldsymbol{x}$ be a vertex of K, and let $P_{\boldsymbol{x}}$ be the affine polyhedral cone formed by the tight constraints $\mathbf{A}$ of K at $\boldsymbol{x}$:

$$P_{\boldsymbol{x}} = \{\boldsymbol{y} + \boldsymbol{x} : \mathbf{A}\boldsymbol{y} \geq \mathbf{0}\}. \tag{3.5}$$

See Figure 3.2 for a visualization.

Proposition 2. *A point $\boldsymbol{y} \in \mathbb{R}^n$ not in K has $C(\boldsymbol{y}) = \boldsymbol{x}$ if and only if $\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ$.*

Proof. If $\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ$, then $(\boldsymbol{y} - \boldsymbol{x})^\intercal(\boldsymbol{z} - \boldsymbol{x}) \leq 0$ for all $\boldsymbol{z} \in P_{\boldsymbol{x}}$. Note that $K \subseteq P_{\boldsymbol{x}}$ so the same holds for all $\boldsymbol{z} \in K$. We have

$$\|\boldsymbol{y} - \boldsymbol{z}\|_2^2 = (\boldsymbol{y} - \boldsymbol{z})^\intercal(\boldsymbol{y} - \boldsymbol{z}) \tag{3.6}$$

$$= (\boldsymbol{y} - \boldsymbol{x})^\intercal(\boldsymbol{y} - \boldsymbol{x}) + 2(\boldsymbol{y} - \boldsymbol{x})^\intercal(\boldsymbol{x} - \boldsymbol{z}) + (\boldsymbol{x} - \boldsymbol{z})^\intercal(\boldsymbol{x} - \boldsymbol{z}). \tag{3.7}$$

The second term is nonnegative because $\boldsymbol{z} \in P_{\boldsymbol{x}}$ and $\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ$, and the third term is nonegative since it is $\|\boldsymbol{x} - \boldsymbol{z}\|_2^2$. Therefore, $\|\boldsymbol{y} - \boldsymbol{x}\|_2 \leq \|\boldsymbol{y} - \boldsymbol{z}\|_2$, so $C(\boldsymbol{y}) = \boldsymbol{x}$.

If $\boldsymbol{y} \notin P_{\boldsymbol{x}}^\circ$, then $\boldsymbol{y}$ is either (i) in the polar cone $P_{\boldsymbol{v}}^\circ$ of some other vertex $\boldsymbol{v}$ of K or (ii) there is some d such that $\boldsymbol{y} \in dH$ for a facet H of K. This is because

$$\{P_{\boldsymbol{v}}^\circ : \text{vertices } \boldsymbol{v}\} \cup \{dH : d \in \mathbb{R} \text{ and facets } H\} = \mathbb{R}^n \tag{3.8}$$

since K is full-dimensional. In case (i), $\boldsymbol{y}$ is closest to $\boldsymbol{v}$ by the first part of the proof. In case (ii), suppose $\boldsymbol{h}$ is the unit normal vector of H, then $\boldsymbol{y} + d\boldsymbol{h} \in H$. By Pythagoras,

$$\|d\boldsymbol{h}\|_2^2 + \|\boldsymbol{y} + d\boldsymbol{h} - \boldsymbol{x}\|_2^2 = \|\boldsymbol{y} - \boldsymbol{x}\|_2^2. \tag{3.9}$$

Thus

$$\|d\boldsymbol{h}\|_2 \leq \|\boldsymbol{y} - \boldsymbol{x}\|_2. \tag{3.10}$$

Therefore $C(\boldsymbol{y}) \neq \boldsymbol{x}$. □

Now, we will show that almost the entire space is closest to some vertex of K. which is mentioned but not proved in [7]. This turns out to be an important lemma in the study of polyhedral complexes [28], but the referenced proof is beyond the scope of this thesis. Here, we present a simpler proof. Let $\mathcal{P} = \bigcup_{\text{vertices } \boldsymbol{v} \text{ of } K} P_{\boldsymbol{v}}^{\circ}$.

Fact 2. *Almost all points in $\mathbb{R}^n$ are in $\mathcal{P}$, in the sense that the set of points not in $\mathcal{P}$ is measure zero.*

Proof. Without loss of generality suppose K contains the origin. Further, suppose K is contained within a ball $S(\mathbf{0}, R)$. Let $\alpha \geq 1$, $\mathbf{vol}(S_\alpha) = \mathbf{vol}(S(\mathbf{0}, \alpha R))$, and $\mathbf{vol}(\mathcal{P}_\alpha) = \mathbf{vol}(\{\boldsymbol{x} \in \mathcal{P} : \boldsymbol{x} \in S(\mathbf{0}, \alpha R)\})$.

An alternative way to view $P_{\boldsymbol{x}}^{\circ}$ is the set of objective vectors $\boldsymbol{c}$ for which the vertex $\boldsymbol{x}$ uniquely solves the linear program $\max_{\boldsymbol{z} \in K} \boldsymbol{c}^\intercal \boldsymbol{z}$. If all the $P_{\boldsymbol{x}}^{\circ}$ are centered at the origin, then $\mathbf{0}$ uniquely solves the LP for all $\boldsymbol{c} \in \mathbb{R}^n$, so $\mathcal{P} = \mathbb{R}^n$. In other words,

$$\frac{\|\boldsymbol{x}\|_2}{\alpha R} = 0 \text{ for any vertex } \boldsymbol{x} \text{ of } K \implies \frac{\mathbf{vol}(\mathcal{P}_\alpha)}{\mathbf{vol}(S_\alpha)} = 1 \tag{3.11}$$

for any $\alpha \geq 1$. Note that the right-hand side decreases monotonically as the left-hand side increases.

Therefore, as $\alpha \to \infty$, for any vertex $\boldsymbol{x}$ of K,

$$\frac{\|\boldsymbol{x}\|_2}{\alpha R} \to 0, \tag{3.12}$$

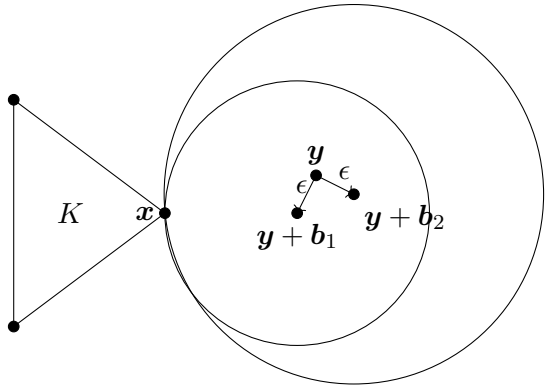


Figure 3.3: Recovering a vertex $\boldsymbol{x}$ of K by perturbing $\boldsymbol{y} \in P_{\boldsymbol{x}}^{\circ}$ along an ϵ-basis and intersecting the resultant circles.

so

$$\frac{\mathbf{vol}(\mathcal{P}_\alpha)}{\mathbf{vol}(S_\alpha)} \to 1. \tag{3.13}$$

□

The takeaway is that if we choose $\boldsymbol{y}$ at random from $\mathbb{R}^n$, it has probability one of being closest to a vertex of K, so we cannot immediately apply Proposition 1. However, we can adapt the same idea to find a point in K. See Figure 3.3 for a visualization.

Proposition 3. *Suppose $\boldsymbol{y} \in P_{\boldsymbol{x}}^{\circ}$ for a vertex $\boldsymbol{x}$ of K. Then we can recover $C(\boldsymbol{y})$ in $O(n)$ distance oracle calls, solving the feasibility problem.*

Proof. Suppose $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ is a basis for $\mathbb{R}^n$ where $\|\boldsymbol{b}_i\|_2 = \epsilon$ for all $i \in \{1, \ldots, n\}$. We need ϵ small enough such that $\boldsymbol{y} + \boldsymbol{b}_i \in P_{\boldsymbol{x}}^{\circ}$ for all i. We will specify this quantity later.

Let $S(\boldsymbol{c}, r)$ denote the $n-1$ dimensional hypersphere of radius r centered at $\boldsymbol{c} \in \mathbb{R}^n$. For each $i \in \{1, \ldots, n\}$ we set $S_i = S(\boldsymbol{y} + \boldsymbol{b}_i, D(\boldsymbol{y} + \boldsymbol{b}_i))$; this takes n calls to the distance oracle. Note $\boldsymbol{x} \in S_i$ for all i. The set

$$S = \bigcap_{i=1}^{n} S_i \tag{3.14}$$

can be computed analytically, for example by intersecting the hyperspheres one at a time and solving for the lower-dimensional sphere [36]. In the final iteration, we are finding the

intersection of two circles, so S contains at most two points and one of them must be $\boldsymbol{x}$. We can test both using the distance oracle. □

While hypersphere intersection is a convenient way to visualize this high-dimensional triangulation, computing the intersection analytically is difficult to formalize. However, it is in fact equivalent to solving a linear system similarly to the method of Proposition 1. Let

$$\mathbf{B} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \text{ and } \boldsymbol{d} = \begin{bmatrix} (d^2 - d_1 + \epsilon^2)/2 \\ (d^2 - d_2 + \epsilon^2)/2 \\ \vdots \\ (d^2 - d_n^2 + \epsilon^2)/2 \end{bmatrix}. \tag{3.15}$$

Since all the $\boldsymbol{b_i}$ are linearly independent, $\mathbf{B}$ is nonsingular. If $\|\boldsymbol{y} + \boldsymbol{b_i} - \boldsymbol{x}\| = d_i$, we have the system of linear equations

$$\mathbf{B}(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{d}. \tag{3.16}$$

and by solving this system, we obtain the unique solution $\boldsymbol{x}$. Note that this system is linear because all the quadratic terms are given by the distance oracle.

3.2.3 Perturbation Distance

We will now analyze the perturbation distance ϵ we need for Proposition 1 and 3. Suppose we are given a hypercube B of radius R containing K; this is similar to the assumption on R for the Ellipsoid Method in Section 1.2.1. Let $U(B)$ denote the uniform distribution over B, sampled by drawing a vector $\boldsymbol{y}$ with $y_i \sim U(0, R)$ for all $i \in \{1, 2, \ldots, n\}$. While we are still developing the general proof, we will show a preliminary result for K a hypercube.

First, we need to analyze how large R needs to be to achieve high probability that $\boldsymbol{y} \sim U(B)$ lies $P_{\boldsymbol{x}}^{\circ}$ for a vertex $\boldsymbol{x}$ of K.

Lemma 2. *Suppose K is a hypercube with side length r, $\boldsymbol{y} \sim U(B)$, and the ratio $\frac{R}{r}$ is constant. Then $R = O(n)$ implies that $\boldsymbol{y}$ lies in $P_{\boldsymbol{x}}^{\circ}$ for a vertex $\boldsymbol{x}$ of K with probability at least $\frac{1}{2}$.*

Proof. Without loss of generality assume $r = 1$. We may also assume K is centered in B, since this minimizes the overall volume

$$\frac{1}{\mathbf{vol}(B)} \sum_{\text{vertices } \boldsymbol{v}} \mathbf{vol}(P_{\boldsymbol{v}}^{\circ}). \tag{3.17}$$

Since K is a hypercube, it has 2^n vertices. Fix a vertex $\boldsymbol{x}$ of K. Then,

$$\begin{aligned}
\Pr[\exists \boldsymbol{v} : \boldsymbol{y} \in P_{\boldsymbol{v}}^{\circ}] &= 2^n \Pr[\boldsymbol{y} \in P_{\boldsymbol{x}}^{\circ}] && (3.18)\\
&= 2^n \frac{\mathbf{vol}(P_{\boldsymbol{x}}^{\circ})}{\mathbf{vol}(B)} && (3.19)\\
&= 2^n \frac{(\frac{R-1}{2})^n}{R^n} && (3.20)\\
&= (\frac{R-1}{R})^n && (3.21)\\
&\leq e^{-\frac{n}{R}}. && (3.22)
\end{aligned}$$

Then $R \geq \frac{n}{\log(2)}$ implies

$$e^{-\frac{n}{R}} \leq \frac{1}{2}. \tag{3.23}$$

□

Then, by sampling $O(\log(n))$ points from $U(B)$, with high probability at least one of them will lie within a polar cone of a vertex of K. We can now calculate ϵ. While we are still working on the complete proof, we will show a preliminary result here.

Proposition 4. *Suppose K is a hypercube with side length r, $\boldsymbol{y} \sim U(B)$, the ratio $\frac{R}{r}$ is constant, and $R = O(n)$. Let $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ be a basis for $\mathbb{R}^n$ where $\|\boldsymbol{b}_i\|_2 = \epsilon$ for all $i \in \{1, \ldots, n\}$. If $\boldsymbol{y} \in P_{\boldsymbol{x}}^{\circ}$ where $\boldsymbol{vol}(P_{\boldsymbol{x}}^{\circ}) \geq \boldsymbol{vol}(P_{\boldsymbol{v}}^{\circ})$ for all vertices $\boldsymbol{v}$ of K, then $\epsilon = O(\frac{1}{2^n})$ implies $\boldsymbol{y} + \boldsymbol{b}_i \in P_{\boldsymbol{x}}^{\circ}$ for all i with probability at least $1 - \frac{1}{n}$.*

Proof. Notice that $P_{\boldsymbol{x}}^{\circ}$ is a polyhedral cone; let $H_1, H_2, \ldots, H_n$ be the facets of $P_{\boldsymbol{x}}^{\circ}$. By

Proposition 2,

$$\Pr[\text{failure}] = \Pr[\exists i : C(\boldsymbol{y} + \boldsymbol{b}_i) \neq \boldsymbol{x}] \tag{3.24}$$

$$= \Pr[\exists i : \boldsymbol{y} + \boldsymbol{b}_i \notin P_{\boldsymbol{x}}^\circ] \tag{3.25}$$

$$\leq \Pr[\exists j : \min_{\boldsymbol{z} \in H_j} \|\boldsymbol{y} - \boldsymbol{z}\|_2 < \epsilon]. \tag{3.26}$$

Fix $k \in \{1, \ldots, n\}$, then by the union bound,

$$\Pr[\exists j : \min_{\boldsymbol{z} \in H_j} \|\boldsymbol{y} - \boldsymbol{z}\|_2 < \epsilon] \leq n \Pr[\min_{\boldsymbol{z} \in H_k} \|\boldsymbol{y} - \boldsymbol{z}\|_2 < \epsilon]. \tag{3.27}$$

Let T be the set of points in $P_{\boldsymbol{x}}^\circ$ at most ϵ away from H_k; formally,

$$T = \{\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ : \|\boldsymbol{y} - \boldsymbol{z}\|_2 < \epsilon \text{ for some } \boldsymbol{z} \in H_k\}. \tag{3.28}$$

Then,

$$n \Pr[\min_{\boldsymbol{z} \in H_k} \|\boldsymbol{y} - \boldsymbol{z}\|_2 < \epsilon] = n \Pr[\boldsymbol{y} \in T] \tag{3.29}$$

$$= n \frac{\mathbf{vol}(T)}{\mathbf{vol}(P_{\boldsymbol{x}}^\circ)}. \tag{3.30}$$

Because K is bounded by a hypercube of side length R, each fundamental vector of P° has length at most R. Thus, T is contained within a hyperprism with $n-1$ sides of length at most R and one side of length ϵ. So, $\mathbf{vol}(T) \leq \epsilon R^{n-1}$. Since K is a hypercube and $\mathbf{vol}(P_{\boldsymbol{x}}^\circ) \geq \mathbf{vol}(P_{\boldsymbol{v}}^\circ)$ for all vertices $\boldsymbol{v}$ of K, by Lemma 2 we have

$$\mathbf{vol}(P_{\boldsymbol{x}}^\circ) \geq \mathbf{vol}(B) \Pr[\boldsymbol{y} \in P_{\boldsymbol{x}}^\circ \text{ when } K \text{ is centered in } B] \tag{3.31}$$

$$= \frac{R^n}{2^{n+1}}. \tag{3.32}$$

Then,

$$n \frac{\mathbf{vol}(T)}{\mathbf{vol}(P_{\boldsymbol{x}}^\circ)} \leq n \frac{\epsilon R^{n-1}}{\frac{R^n}{2^{n+1}}} \tag{3.33}$$

$$= n \frac{\epsilon 2^{n+1}}{R}. \tag{3.34}$$

Then by choosing $\epsilon \leq \frac{R}{n^2 2^{n+1}} = O(\frac{1}{2^n})$, the probability of failure is at most $\frac{1}{n}$. □

The approach above is incomplete because we would then need to analyze the probability we land in the largest cone specifically, and it can be exponentially small. One possible direction would be to amortize over all polar cones and show that we are likely to land in a "large enough" cone; this makes sense because the probability we land in a certain cone is proportionate to its volume.

Our overall algorithm would then be to set $R = O(n)$ and sample $O(\log(n))$ points from $U(B)$ to guarantee landing in a polar cone with high probability, then for each point draw an ϵ-basis using Proposition 4 and solve the linear system of Proposition 3. This algorithm has an oracle complexity of $O(n \log(n))$, a factor of $\log(\frac{R}{\delta})$ better than the lower bound on oracle complexity of separation oracle-based methods [29], which the current state-of-the-art method achieves [21]. Here, δ is the approximation parameter in the weak definition of the feasibility problem (see Section 1.1.2).

Note that this algorithm solves the *strong* linear feasibility problem. General convex sets are smooth and do not necessarily have facets or vertices, so we will not be able to find a viable perturbation distance ϵ. However, we may still be able to solve the *weak* convex feasibility problem. To do so, we would adapt Proposition 3; in short, each center would not be closest to the same point, but those closest points must not be too far from each other. So, analyzing the approximate intersection of all hyperspheres may be enough to guarantee finding a nearly feasible point. We predict that solving the weak problem would introduce a $\log(\frac{1}{\delta})$ into the oracle complexity, but our algorithm would still improve upon the separation oracle method by a factor of $\log(R)$. We leave this to future work.

One remark is that our algorithm does not enable solving the feasibility problem without loss of generality. In particular, we require $R = O(n)$ for sampling. This may not be possible in application scenarios such as the interactive learning example in Section 3.1. In that case, classifiers are constrained to the domain $[0, 1]^n$ and if we set $R = O(n)$ our oracle may not be able to interpret the input. Thus, it would be useful to develop an algorithm that does not require sampling and works whether the proposed point is closest to a facet, vertex, or something in-between. We hypothesize that the linear system of Proposition 3 can be

adapted for when the proposed point is closest to a lower-dimensional face F if we have a *subset selection oracle*, which returns the dimensionality of F and a basis for its subspace. In this case, we can use the given basis as the perturbation vector $\boldsymbol{b}$ and solve the linear system to obtain a point on F. We leave a more thorough investigation to future work.

3.3 The Oracle Power Hierarchy

Besides its application to developing efficient algorithms for feasibility when separation is difficult, the distance oracle is interesting to study for theoretical reasons. In particular, why does distance enable a faster algorithm than separation, and what makes some oracles more powerful than others? Here, we develop a framework for analyzing oracle power and show how the distance oracle and other work fit into the system; the overall goal of this line of research is to characterize the tradeoff between quality of information given by the oracle and the complexity of solving the feasibility problem.

The Direction Oracle

As a warm-up, we consider an especially powerful oracle called the direction oracle. We define the Direction problem (DIR) as follows: Given $\boldsymbol{y} \in \mathbb{R}^n$, either assert that $\boldsymbol{y} \in K$ or find the unit-length vector $\boldsymbol{z}$ in the direction of the closest point in K to $\boldsymbol{y}$. Formally, if $C(\boldsymbol{y}) = \arg\min_{\boldsymbol{x} \in K} \|\boldsymbol{y} - \boldsymbol{x}\|_2$, then $\boldsymbol{z} = \frac{C(\boldsymbol{y})}{\|C(\boldsymbol{y})\|_2}$. A direction oracle solves DIR.

Since K is contained in a hypercube of side length R, we can use the direction oracle to binary search over the line $[\boldsymbol{y}, \boldsymbol{z}R]$. We query the oracle at each step to determine which side of the interval to recurse on. Thus, the direction oracle solves the weak feasibility problem in $O(\log \frac{R}{\epsilon})$ queries, where ϵ is the approximation parameter (see Section 1.1.2).

The low oracle complexity of this algorithm implies that the direction oracle is more powerful than the distance oracle and separation oracle; in particular, the information given by direction is in some sense more effective than distance or separation. In this case, there is a simple reason: the direction oracle essentially reduces feasibility to a one-dimensional

Reference	Oracle	Oracle Complexity	Additional Information
Ours	Direction	$O(\log(\frac{R}{\epsilon}))$	R
[3, 10]	Quantum	$\Omega(n)$	R
Ours	Distance*	$O(n\log(n))$	R
[18]	Separation	$\Omega(n\log(\frac{nR}{\epsilon}))$	R
[5]	Index-Furthest†	Polynomial	R, m
[5]	Index-Worst	Exponential	N/A
[18]	Membership	Exponential	N/A

Table 3.1: The oracle power hierarchy for the convex feasibility problem. Each oracle is stronger than all those below it. (*) The analysis of the distance oracle is incomplete, but in Section 3.2 we conjecture it can solve the linear problem in $O(n\log(n))$ queries. (†) The index oracle can only solve the linear problem.

problem. However, this deduction is more elusive when analyzing more complicated oracles.

Quantifying Oracle Power

Measuring the relative power of certain oracles is especially interesting because there are multiple notions of efficiency. In particular, two important characteristics are (1) the complexity of the oracle method for solving the feasibility problem, and (2) the conditions under which the oracle can solve the feasibility problem, in terms of necessary information. For example, the separation oracle solves the feasibility problem in $\Omega(n\log(\kappa))$ queries and requires the additional information of an outer radius R.

We will formally define what it means for an oracle to be stronger than another. Suppose $\mathcal{A}$ and $\mathcal{B}$ are oracles which solve the convex feasibility problem in $T_{\mathcal{A}}$ and $T_{\mathcal{B}}$ queries respectively. Furthermore, suppose they require sets of additional information $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$ respectively. Then, $\mathcal{A}$ is stronger than $\mathcal{B}$ for solving the convex feasibility problem if and only if $T_{\mathcal{A}} \leq T_{\mathcal{B}}$ and $S_{\mathcal{A}} \subseteq S_{\mathcal{B}}$. If $\mathcal{A}$ is stronger than $\mathcal{B}$, we say $\mathcal{B}$ is weaker than $\mathcal{A}$.

The definitions are similar for the linear feasibility problem and the approximate convex feasibility problem. Notice that we do not consider runtime complexity in these definitions, only oracle complexity, which allows us to directly measure oracle power without implementation considerations. Additionally, it is difficult to compare oracles which require disjoint sets of information – they are not necessarily stronger or weaker than one another, because

they play by different rules. Finally, it is possible that the most efficient algorithm using $\mathcal{B}$ is to reduce to $\mathcal{A}$ and run the algorithm for $\mathcal{A}$; in this case, $\mathcal{A}$ is still considered stronger by the definition because the reduction cannot be faster than constant time.

See Table 3.1 for an organization of the distance and direction oracles, as well as other both classic and recent oracles, into our oracle power framework.

One novel application of this framework is that we quantify exactly how the index oracle is weaker than the others. The goal of [5] was to design a "weak" oracle, but they did not quantify this rigorously, instead justifying it with the intuition that returning constraint indices seems to be weaker than a separating hyperplane. Here, we provide reasoning for calling the index oracle weak. In particular, it requires knowledge of the number of constraints m of the polytope in addition to an outer radius R, while the separation oracle only requires the latter. Additionally, the index oracle can only solve the linear problem.

3.4 Future Work

This research has generated more questions than answers. Here, we briefly describe some interesting open problems motivated by the distance oracle. Answers to these questions would both increase the comprehensiveness of our work and develop a deeper understanding of the machinery behind oracle methods.

1. We require that the distance oracle return perfectly accurate distances. However, this is often unreasonable, for numerical reasons as well as for real-world applications. While approximation algorithms have been developed for the separation oracle (see Section 2.3.2), it is unclear whether approximate distances would enable approximately solving the feasibility problem or break the algorithm entirely.

2. One way to interpret the distance oracle is that it returns the distance from the query point to the most violated constraint. This leads to a fascinating connection between the distance oracle and the index oracle [5], which returns the *index* of the most violated constraint. For either oracle, returning an answer relative to an

arbitrary constraint does not work [5]. It would be interesting to find a nontrivial reduction between these two oracles and understand what, if anything, is special about leveraging the most violated constraint. Which other oracles utilize the most violated constraint?

3. Both the distance oracle and direction oracle are defined in terms of Euclidean distances. Is there a generalization from the ℓ_2-norm to the ℓ_p-norm? What about statistical distance for the interactive learning application from Section 3.1? More generally, which classes of metrics allow the distance oracle to efficiently optimize?

4. Are there any other applications where separation oracles are difficult to find that may motivate new types of alternative oracles? As a corollary, can we characterize the scenarios where separation oracles would not be the first choice of oracle method?

Bibliography

[1] Jacob Abernethy and Elad Hazan. "Faster Convex Optimization: Simulated Annealing with an Efficient Universal Barrier". In: *33rd International Conference on Machine Learning (ICML 2016)*. URL: https://arxiv.org/abs/1507.02528.

[2] Oleg Alexandrov. *Polar Cone*. URL: https://commons.wikimedia.org/w/index.php?curid=2099628.

[3] Joran van Apeldoorn et al. "Convex Optimization with Quantum Oracles". In: *Quantum* (2020). URL: https://arxiv.org/abs/1809.00643.

[4] David S. Atkinson and Pravin M. Vaidya. "A Cutting Plane Algorithm for Convex Programming that Uses Analytic Centers". In: *Mathematical Programming* (1995). URL: https://link.springer.com/article/10.1007/BF01585551.

[5] Xiaohui Bei, Ning Chen, and Shengyu Zhang. "Solving Linear Programming with Constraints Unknown". In: *42nd International Colloquium on Automata, Languages, and Programming (ICALP 2015)*. URL: https://arxiv.org/abs/1304.1247.

[6] Demitris Bertsimas and Santosh Vempala. "Solving Convex Programs by Random Walks". In: *Journal of the ACM* (2004). URL: https://web.mit.edu/dbertsim/www/papers/Optimization/Solving%5C%20Convex%5C%20Programs%5C%20by%5C%20Random%5C%20Walks.pdf.

[7] Nicolas Bonifas et al. "On sub-determinants and the diameter of polyhedra". In: *28th ACM Symposium on Computational Geometry (SOCG 2012)*. URL: https://arxiv.org/abs/1108.4272.

[8] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. 2009. URL: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf.

[9] Sébastian Bubeck. *Convex Optimization: Algorithms and Complexity*. 2015. URL: http://sbubeck.com/Bubeck15.pdf.

[10] Shouvanik Chakrabarti et al. "Quantum Algorithms and Lower Bounds for Convex Optimization". In: *Quantum* (2020). URL: https://arxiv.org/abs/1809.01731.

[11] Henry Cohn et al. "Group-theoretic algorithms for matrix multiplication". In: *46th IEEE Symposium on Foundations of Computer Science (FOCS 2005)*. URL: https://arxiv.org/abs/math/0511460.

[12] James Demmel, Ioana Dumitriu, and Olga Holtz. "Fast Linear Algebra is Stable". In: *Numerische Mathematik*. 2007. URL: https://arxiv.org/abs/math/0612264.

[13] Shaddin Dughmi. "Consequences of the Ellipsoid Algorithm". University of Southern California Viterbi School of Engineering, CSCI 675: Convex and Combinatorial Optimization. 2019. URL: https://viterbi-web.usc.edu/~shaddin/cs675fa19/slides/ellipsoid_consequences.pdf.

[14] Shaddin Dughmi. "Linear Programming". University of Southern California Viterbi School of Engineering, CSCI 675: Convex and Combinatorial Optimization. 2019. URL: https://viterbi-web.usc.edu/~shaddin/cs675fa19/slides/LP.pdf.

[15] Shaddin Dughmi. "The Ellipsoid Algorithm". University of Southern California Viterbi School of Engineering, CSCI 675: Convex and Combinatorial Optimization. 2019. URL: https://viterbi-web.usc.edu/~shaddin/cs675fa19/slides/ellipsoid.pdf.

[16] Shaddin Dughmi and Haifeng Xu. "Algorithmic Bayesian Persuasion (arXiv v2)". In: *48th ACM Symposium on Theory of Computing (STOC 2016)*. URL: https://arxiv.org/abs/1503.05988v2.

[17] Ehsan Emamjomeh-Zadeh and David Kempe. "A General Framework for Robust Interactive Learning". In: *31st Conference on Neural Information Processing Systems (NeurIPS 2017)*. URL: https://arxiv.org/abs/1710.05422.

[18] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. 1993.

[19] Anupam Gupta. "Lecture 17: The Ellipsoid Algorithm". Carnegie Mellon University School of Computer Science, 15-850: Advanced Algorithms. 2017. URL: https://www.cs.cmu.edu/~anupamg/advalgos17/scribes/lec16.pdf.

[20] Klaus Jansen. "Approximate Strong Separation with Application in Fractional Graph Coloring and Preemptive Scheduling". In: *19th International Symposium on Theoretical Aspects of Computer Science (STACS 2002)*. URL: https://www.sciencedirect.com/science/article/pii/S0304397502008290.

[21] Haotian Jiang et al. "An Improved Cutting Plane Method for Convex Optimization, Convex-Concave Games and its Applications". In: *52nd ACM Symposium on Theory of Computing (STOC 2020)*. URL: https://arxiv.org/abs/2004.04250.

[22] Adam Tauman Kalai and Santosh Vempala. "Simulated Annealing for Convex Optimization". In: *Mathematics of Operations Research* (2006). URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/2006-Simulated_Annealing_for_Convex_Optimization.pdf.

[23] L. G. Khachiyan. "A Polynomial Algorithm in Linear Programming". In: *Soviet Mathematics* (1979).

[24] L. G. Khachiyan, S. P. Tarasov, and I. I. Erlikh. "The Method of Inscribed Ellipsoids". In: *Soviet Mathematics* (1988).

[25] Yin Tat Lee, Aaron Sidford, and Santosh S. Vempala. "Efficient Convex Optimization with Membership Oracles". In: *31st Conference On Learning Theory (COLT 2018)*. URL: https://arxiv.org/abs/1706.07357.

[26] Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. "A Faster Cutting Plane Method and its Implications for Combinatorial and Convex Optimization". In: *56th IEEE Symposium on Foundations of Computer Science (FOCS 2015)*. URL: https://arxiv.org/abs/1508.04874.

[27] A. Yu Levin. "On an Algorithm for the Minimization of Convex Functions". In: *Soviet Mathematics* (1965).

[28] Ricky Liu. "Polyhedral Complexes". North Carolina State University, MA 724: Combinatorics II. 2019. URL: https://riliu.math.ncsu.edu/724/notesse6.html.

[29] A. S. Nemirovskii and D. B. Yudin. *Problem Complexity and Method Efficiency in Convex Optimization*. 1983. URL: hhttps://www2.isye.gatech.edu/~nemirovs/Nemirovskii_Yudin_1983.pdf.

[30] Ryan O'Donnell. "Lecture 8: Ellipsoid Algorithm". Carnegie Mellon University School of Computer Science, 15-859(E): Linear and Semi-definite Programming. 2011. URL: http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15859-f11/www/notes/lecture08.pdf.

[31] Tilman Piesk. *Polar Cube*. URL: https://commons.wikimedia.org/w/index.php?curid=65865974.

[32] Naum Z. Shor. "Utilization of the Operation of Space Dilatation in the Minimization of Convex Functions". In: *Cybernetics* (1970).

[33] Pravin M. Vaidya. "A New Algorithm for Minimizing Convex Functions over Convex Sets". In: *30th IEEE Symposium on Foundations of Computer Science (FOCS 1989)*. URL: https://www.computer.org/csdl/pds/api/csdl/proceedings/download-article/12OmNvDqsGW/pdf.

[34] Matthew S. Weinberg. "Algorithms for Strategic Agents". PhD thesis. Massachusetts Institute of Technology, 2014.

[35] Matthew S. Weinberg. "Oracles, Ellipsoid Method, and their Uses in Convex Optimization". Princeton University Department of Computer Science, COS 521: Advanced Algorithm Design. 2017. URL: https://www.cs.princeton.edu/~smattw/Teaching/521fa17lec15.pdf.

[36] Eric W. Weisstein. *Sphere-Sphere Intersection*. From MathWorld – A Wolfram Web Resource. URL: https://mathworld.wolfram.com/Sphere-SphereIntersection.html.

[37] D. B. Yudin and A. S. Nemirovskii. "Evaluation of the Information Complexity of Mathematical Programming Problems". In: *Matekon* (1976).