Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time

Daniel A. Spielman*
Department of Mathematics
M.I.T.
Cambridge, MA 02139
spielman@mit.edu

Shang-Hua Teng†
Akamai Technologies Inc. and
Department of Computer Science
University of Illinois at Urbana-Champaign
steng@cs.uiuc.edu

ABSTRACT

We introduce the *smoothed analysis of algorithms*, which is a hybrid of the worst-case and average-case analysis of algorithms. Essentially, we study the performance of algorithms under small random perturbations of their inputs. We show that the shadow-vertex simplex algorithm has polynomial *smoothed complexity*.

1. INTRODUCTION

The Analysis of Algorithms community has been challenged by the existence of remarkable algorithms that are known by scientists and engineers to work well in practice, but whose theoretical analyses are negative or inconclusive. The root of this problem is that algorithms are usually analyzed in one of two ways: by worst-case or average-case analysis. Worst-case analysis can improperly suggest that an algorithm will perform poorly by examining its performance under incredibly contrived circumstances that may never occur in practice. On the other hand, while many algorithms perform unusually well on random inputs considered in average-case analysis, randomly generated inputs often bear little resemblance to those actually encountered in practice.

We propose an analysis that we call smoothed analysis that can help explain the success of many algorithms that both worst-case and average case cannot. In smoothed analysis, we measure the performance of an algorithm under slight random perturbations of arbitrary inputs. In particular, we consider Gaussian perturbations of inputs to algorithms that take real and complex inputs, and we measure the running time of algorithms in terms of the input size and the variance of the Gaussian perturbations.

We show that the shadow-vertex simplex algorithm has polynomial smoothed complexity. The simplex algorithm is the classic example of an algorithm that is known to perform well in practice but which takes exponential time in the worst case. In the late 1970's and early 1980's the simplex algorithm was shown to converge in expected polynomial time on various distributions of inputs. However, one cannot infer from these analyses that the algorithm should perform well on the distributions encountered in practice.

For every matrix A with entries of absolute value at most 1, every vector $\boldsymbol{z}$, and every vector $\boldsymbol{y}$ whose entries are 1 or -1, we show that the simplex algorithm using the shadow-vertex pivot rule almost always takes time polynomial in $1/\sigma$ and the sizes of A and $\boldsymbol{z}$ to solve

$$\begin{aligned} &\text{maximize} && \boldsymbol{z}^T x \\ &\text{subject to} && (A + \sigma G)\boldsymbol{x} \leq \boldsymbol{y}. \end{aligned} \tag{1}$$

We remark that these restrictions do not change the family of linear programs expressible, and that it is simple to transform any linear program into one of this form. Moreover, if A is well-scaled, then the solution to this program is an approximation to the solution of the original.

This extended abstract merely outlines the proofs of our results. The complete paper may be found at `http://math.mit.edu/~spielman/simplex`.

1.1 Background

It is difficult to overstate the importance of linear programming to optimization. Linear programming problems arise in innumerable industrial contexts. Moreover, linear programming is often used as a fundamental step in other optimization algorithms. In a linear programming problem, one is asked to maximize or minimize a linear function over a polyhedral region. The general form of a linear program is

$$\begin{aligned} &\text{maximize} && \boldsymbol{z}^T \boldsymbol{x} \\ &\text{subject to} && A\boldsymbol{x} \leq \boldsymbol{y}, \end{aligned} \tag{2}$$

where A is a m-by-d matrix, $\boldsymbol{z}$ is a d-vector, and $\boldsymbol{y}$ is an m-vector.

Perhaps one reason that we see so many linear programs is that we can solve them efficiently. In 1947, Dantzig [7]

*Partially supported by an Alfred P. Sloan Foundation Fellowship, and NSF CAREER award CCR: 9701304. Much of this paper was written during a Junior Faculty Research Leave sponsored by the M.I.T. School of Science

†Partially suppoted by an Alfred P. Sloan Foundation Fellowship, and NSF grant CCR: 99-72532

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.
Copyright 2001 ACM 1-58113-349-9/01/0007 ...$5.00.

introduced the simplex method, which was the first practical approach to solving linear programs and which remains widely used today. To state it roughly, the simplex algorithm proceeds by walking from one vertex to another of the polyhedron defined by the inequalities in (2). At each step, it walks to a vertex that is better with respect to the objective function. The algorithm will either determine that the constraints are unsatisfiable, determine that the objective function is unbounded, or reach a vertex from which it cannot make progress, which necessarily optimizes the objective function.

Two provably polynomial time algorithms for linear programming are known: the ellipsoid algorithm [17] and the interior-point method [16]. In spite of the good practical performance of some interior-point methods, the simplex method remains the most popular method for solving linear programs. However, there has been no satisfactory theoretical explanation of its excellent performance. A fascinating approach to understanding the performance of the simplex algorithm has been the attempt to prove that there always exists a short walk from each vertex to the optimal vertex. The Hirsch conjecture states that there should always be a walk of length $m - d$. Significant progress on this conjecture was made by Kalai and Kleitman [15], who proved that there always exists a walk of length at most $m^{\log_2 d+2}$. However, the existence of such a short walk does not imply that the simplex method will find it.

A simplex method is not completely defined until one specifies its *pivot rule*—the method by which it decides which vertex to walk to when it has many to choose from. There is no deterministic pivot rule under which the simplex method is known to take a sub-exponential number of steps. In fact, for almost every deterministic pivot rule there is a family of polytopes on which it is known to take an exponential number of steps[18, 22, 11, 10, 3, 13]. The best present analysis of randomized pivot rules shows that they take expected time $m^{O(\sqrt{d})}$[14, 19], which is quite far from the polynomial complexity observed in practice. This inconsistency between the exponential worst-case behavior of the simplex algorithm and its everyday practicality leave us wanting a reasonable theoretical analysis.

Various average-case analyses of the simplex algorithm have been performed. Most relevant to this paper is the analysis of Borgwardt [5, 6] who proved that the simplex algorithm with the shadow vertex pivot rule runs in expected polynomial time for polytopes whose constraints are drawn independently from spherically symmetric distributions (*e.g.* Gaussian distributions centered at the origin). Independently, Smale [25, 24] proved bounds on the expected running time of the related parametric Lemke algorithm on random instances of the linear complementarity problem (LCP) chosen from a particular spherically-symmetric distribution. Another model of random linear programs was studied in a parallel line of research initiated independently in unpublished manuscripts by Haimovich (Columbia, 1983) and Adler (Berkeley, 1983). Their works considered linear programs with arbitrary defining matrices in which the only randomness appears in the random choice of the directions of the inequalities. They proved that parametric simplex methods would take expected linear time to solve phase II linear programming problems. Complete algorithms for linear programming under this model were analyzed by Todd [26] and Adler and Megiddo [2] who proved quadratic bounds on the expected time taken by the lexicographic Lemke algorithm to solve LCP problems derived from linear programming problems. Further average-case results were obtained in [1] and [20].

While these average-case analyses are significant accomplishments, they do not explain the performance of the simplex method in practice. Problem instances encountered in practice may have little resemblance to those generated at random. Moreover, it is now well-understood that random combinatorial objects have many special properties. Edelman [8] writes on this point:

> What is a mistake is to psychologically link a random matrix with the intuitive notion of a "typical" matrix or the vague concept of "any old matrix."

1.2 Smoothed Analysis of Algorithms

We introduce the *smoothed analysis of algorithms* in the hope that it will succeed in explaining the good practical performance of many algorithms for which worst-case and average-case analysis have failed. Our first application of the smoothed analysis of algorithms will be to the simplex algorithm. We will consider the running time of the simplex algorithm on inputs of the form

$$\begin{aligned} &\text{maximize} && \boldsymbol{z}^T\boldsymbol{x} \\ &\text{subject to} && (A + \sigma G)\boldsymbol{x} \leq \boldsymbol{y}, \end{aligned} \tag{3}$$

where we let A be arbitrary and G be a matrix of independently chosen Gaussian random variables of mean 0 and variance 1. If we let σ go to 0, then we obtain the worst-case complexity of the simplex algorithm; whereas, if we let σ be so large that σG swamps out A, we obtain the average-case analyzed by Borgwardt. By choosing polynomially small σ, this analysis combines advantages of worst-case and average-case analysis.

In a smoothed analysis of an algorithm, we assume that the inputs to the algorithm are subject to slight random perturbations, and we measure the complexity of the algorithm in terms of the input size and the variance of the perturbations. If an algorithm has low smoothed complexity, then one should expect it to work well in practice since most real-world problems are generated from data that is inherently noisy. Another way of thinking about smoothed complexity is to say that if an algorithm has low smoothed complexity, then one must be unlucky to choose an input instance on which it performs poorly.

We now provide some definitions for the smoothed analysis of algorithms that take real or complex inputs. For an algorithm A and input x, let

$$C(A, x)$$

be a complexity measure of A on input x. Let X be the domain of inputs to A, and let X_n be the set of inputs of size n. The size of an input can be measured in various ways. Standard measures are the number of real variables contained in the input and the sums of the bit-lengths of the variables. Using this notation, one can say that A has worst-case C-complexity $f(n)$ if

$$\max_{x \in X_n} (C(A, x)) = f(n).$$

Given a family of distributions μ_n on X_n, we say that A has

average-case C-complexity $f(n)$ under μ if

$$\mathbf{E}_{x \stackrel{\mu_n}{\leftarrow} X_n}[C(A,x)] = f(n).$$

Similarly, we say that A has *smoothed C-complexity* $f(n,\sigma)$ if

$$\max_{x \in X_n} \mathbf{E}_g[C(A, x + \sigma \max(x) g)] = f(n,\sigma),$$

where g is a vector of Gaussian random variables of mean 0 and variance 1 and $\max(x)$ is the maximum entry in x. Even if we cannot bound the expectation over the perturbations, we can still obtain computationally meaningful results for an algorithm by proving that it has ϵ-smoothed-complexity $f(n,\sigma)$, by which we mean that

$$\forall_{x \in X_n} \mathbf{Pr}_g[C(A, x + \sigma \max(x) g) \leq f(n,\sigma)] \geq 1 - \epsilon.$$

One criticism of *smoothed complexity* as defined above is that the Gaussian perturbations destroy any zero-structure that the problem had, as it will replace the zeros with small values. One can refine the model to fix this problem by studying *relative perturbations*. Under a relative perturbation, an input is mapped to a constant multiple of itself. For example, a reasonable definition would be to map each variable by

$$x \mapsto x(1 + \sigma g),$$

where g is a Gaussian random variable of mean zero and variance 1. Thus, each number is usually mapped to one of similar magnitude, and zero is always mapped to zero. When we measure smoothed complexity under relative perturbations, we call it *relative smoothed complexity*. Smooth complexity as defined above can be called *absolute smoothed complexity* if clarification is necessary.

In Section 5, we present some further generalizations of these definitions.

1.3 The Shadow Vertex Pivot Rule

Gass and Saaty [9] introduced the shadow vertex algorithm as a parametric approach to solving linear programs. Starting at a vertex $\boldsymbol{x}$, the algorithm chooses some objective function, $\boldsymbol{t}$, that is optimized by $\boldsymbol{x}$. The algorithm then slowly transforms $\boldsymbol{t}$ into the objective function of the linear program, $\boldsymbol{z}$, finding the vertex that optimizes each intermediate objective function. By remembering $\boldsymbol{t}$, this provides a pivot rule, known as the Gass-Saaty rule. The vertices encountered during the resulting walk are contained in the set of vectors that optimize functions in the span of $\boldsymbol{t}$ and $\boldsymbol{z}$. The algorithm gets its name from the fact that if one projects the polytope to the plane spanned by $\boldsymbol{t}$ and $\boldsymbol{z}$, then these are pre-images of the corners of the resulting polygon. Borgwardt's analysis used the shadow vertex algorithm.

The shadow-vertex pivot rule is best described through a polar formulation. To understand this formulation, first recall what it means for a point $\boldsymbol{x}$ to be a solution to

$$\begin{aligned} &\text{maximize} && \boldsymbol{z}^T \boldsymbol{x} \\ &\text{subject to} && \langle \boldsymbol{a}_i | \boldsymbol{x} \rangle \leq y_i, \text{ for } 1 \leq i \leq m \end{aligned} \tag{4}$$

CLAIM 1.1. *A point $\boldsymbol{x}$ is a solution to (4) if and only if*

1. *$\boldsymbol{x}$ is feasible, that is $\langle \boldsymbol{a}_i | \boldsymbol{x} \rangle \leq y_i$, for $1 \leq i \leq m$.*
2. *$\boldsymbol{x}$ is extremal: there exists a set B of inequalities such that $\langle \boldsymbol{a}_i | \boldsymbol{x} \rangle = \boldsymbol{y}_i$, for $\{\boldsymbol{a}_i : i \in B\}$, and*
3. *$\boldsymbol{x}$ is optimal: $\boldsymbol{z}$ can be expressed as*

$$\boldsymbol{z} = \sum_{i \in B} \alpha_i \boldsymbol{a}_i, \text{ with } \alpha_i \geq 0.$$

The polar of vertex $\boldsymbol{x}$ is the convex hull of $\{\boldsymbol{a}_i : i \in B\}$. For points in general position, B is a set of d linearly independent inequalities and their convex hull is a simplex of co-dimension 1. Condition (3) says that $\boldsymbol{z}$ lies in the cone from the origin through this simplex.

DEFINITION 1.2. *For vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$, and a unit vector $\boldsymbol{z}$, we let*

$$\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$$

denote the simplex $\mathbf{Simplex}(\{\boldsymbol{a}_i : i \in B\})$*, where B is as in Claim 1.1. If the program (4) is unbounded or infeasible, then we let* $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ *be the empty set.*

If the origin is feasible, then $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ is a facet of the convex hull of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ and every facet of the convex hull optimizes some objective function. If some $y_i < 0$, then we do not obtain as nice a characterization of $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$.

For linear programs in which the origin is feasible, the shadow-vertex simplex algorithm can be seen to walk along the exterior of the convex hull of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$. We assume that it begins knowing a facet of the convex hull (how one finds a starting facet or handles a program with infeasible origin is discussed in Section 4). It then chooses some point inside this facet, and forms the unit vector in the direction of this point, which we will call $\boldsymbol{t}$. The algorithm then examines the directions that are encountered as it transforms $\boldsymbol{t}$ into $\boldsymbol{z}$, and tracks the facets encountered along the way. To make this formal let

$$\boldsymbol{t}_\beta = \frac{\beta \boldsymbol{t} + (1-\beta)\boldsymbol{z}}{\|\beta \boldsymbol{t} + (1-\beta)\boldsymbol{z}\|};$$

so, $\boldsymbol{t}_1 = \boldsymbol{t}$ and $\boldsymbol{t}_0 = \boldsymbol{z}$. As β goes from 1 to 0, the shadow-vertex simplex algorithm encounters the facets

$$\mathbf{optSimp}_{\boldsymbol{t}_\beta;\boldsymbol{y}}(a_1, \ldots, a_m).$$

Of course, the algorithm does not actually modify β continuously, but rather computes each facet along the path from the facet previously encountered.

For general linear programs, the simplices encountered by the shadow vertex algorithm do not necessarily lie on the convex hull of the $\boldsymbol{a}_i$s. So, we introduce the notation

$$\mathbf{optSimps}_{\boldsymbol{z},\boldsymbol{t};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \stackrel{\text{def}}{=} \bigcup_{\boldsymbol{v} \in \mathbf{Span}(\boldsymbol{z},\boldsymbol{t})} \mathbf{optSimp}_{\boldsymbol{v};\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m),$$

which captures the set of simplices the shadow vertex algorithm could encounter during its walk.

1.4 Our Results

We consider linear programming problems of the form

$$\begin{aligned} &\text{maximize} && \boldsymbol{z}^T \boldsymbol{x} \\ &\text{subject to} && \langle \boldsymbol{a}_i | \boldsymbol{x} \rangle \leq y_i, \text{ for } 1 \leq i \leq m \end{aligned} \tag{5}$$

where each $y_i \in \{1, -1\}$, and $\|\boldsymbol{a}_i\| \leq 1$. We remark that these restrictions do not change the family of linear programs expressible, and that it is simple to transform any

linear program into one of this form: first map $\boldsymbol{x} \mapsto \boldsymbol{x} + \boldsymbol{v}$ for some vector $\boldsymbol{v}$ so that none of the y_is are zero, and then map $\boldsymbol{x} \mapsto \boldsymbol{x} \max_i \|\boldsymbol{a}_i\|$.

We study the perturbations of these programs given by

$$\begin{array}{ll} \text{maximize} & \boldsymbol{z}^T\boldsymbol{x} \\ \text{subject to} & \langle \boldsymbol{b}_i | \boldsymbol{x} \rangle \leq y_i, \text{ for } 1 \leq i \leq m \end{array} \tag{6}$$

where

$$\boldsymbol{b}_i = \boldsymbol{a}_i + \boldsymbol{g}_i$$

and $\boldsymbol{g}_i$ is a Gaussian random vector of variance σ^2 centered at the origin. Thus, $\boldsymbol{b}_i$ is a Gaussian random vector of variance σ^2 centered at $\boldsymbol{a}_i$. Throughout the paper, we will assume $0 \leq \sigma \leq 1$.

Our first result, Theorem 3.1, says that, for every program of form (5) and for every vector $\boldsymbol{t}$, the expected size of $\mathbf{optSimps}_{\boldsymbol{z},\boldsymbol{t},\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ is polynomial in m, the dimension, and $1/\sigma$. To state this in terms of smoothed complexity, let $C(S)$ count the number of items in the set S. Let $X_{d,m}$ be the set of inputs in d-dimensions with m constraints:

$$X_{d,m} = \left\{ (\boldsymbol{t}, \boldsymbol{z}, \boldsymbol{y}; \boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \middle| \boldsymbol{t} \in \mathbb{R}^d, \boldsymbol{z} \in \mathbb{R}^d, \ \boldsymbol{y} \in \{0,1\}^m, \right. \\ \left. \boldsymbol{a}_i \in \mathbb{R}^d, \text{ and } \|\boldsymbol{a}_i\| \leq 1, \text{ for } 1 \leq i \leq m \right\}.$$

Then, our result says that the smoothed C-complexity of $\mathbf{optSimps}_{\boldsymbol{t},\boldsymbol{z},\boldsymbol{y}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ is polynomial in m, d, and $1/\sigma$, where we only apply perturbations to the $\boldsymbol{a}_i$s. This roughly corresponds to analyzing Phase II of a simplex algorithm.

In Section 4, we describe a simple two-Phase shadow-vertex simplex algorithm that almost always solves the linear program (6) in time polynomial in m, the dimension, and $1/\sigma$. If we let C measure the running time of an algorithm, then Theorem 4.2 shows that the ϵ-smoothed C-complexity of this simplex algorithm is polynomial in m, d, $1/\sigma$ and $1/\epsilon$ for every ϵ.

To justify perturbing linear programs, we show in Section 2.1 that if the polytope corresponding to the equations (5) contains a ball of radius r and is contained in a co-centric ball of radius R, then for $\sigma < r/dR$ the solution to (6) is an approximation to the solution of (5), with high probability.

2. GAUSSIAN PERTURBATIONS

In this section, we formally describe how we perturb linear programming problems and explore how these perturbations change polytopes.

Recall that a d dimensional Gaussian distribution with variance σ^2 centered at a point $\boldsymbol{a}$ is defined by the probability density function

$$\mu(\boldsymbol{b}) = \left(1/\sqrt{2\pi\sigma}\right)^d e^{-\|\boldsymbol{a}-\boldsymbol{b}\|^2/2\sigma^2}.$$

Throughout the paper, we will assume $0 \leq \sigma \leq 1$.

For a polytope $P_{\boldsymbol{a}}$ given by the equations

$$\langle \boldsymbol{a}_i | \boldsymbol{x} \rangle \leq y_i,$$

where $y_i \in \{1, -1\}$ and $\|a_i\| \leq 1$, we will examine polytopes $P_{\boldsymbol{b}}$ defined by

$$\langle \boldsymbol{b}_i | \boldsymbol{x} \rangle \leq y_i,$$

where $\boldsymbol{b}_i = \boldsymbol{a}_i + \boldsymbol{g}_i$ and $\boldsymbol{g}_i$ is a Gaussian random vector of variance σ^2 centered at the origin. We remark that $P_{\boldsymbol{b}}$ is a simple polytope with probability 1. That is, each vertex of $P_{\boldsymbol{b}}$ is the intersection of exactly d hyperplanes.

We will make frequent use of the following fundamental fact about Gaussian distributions.

LEMMA 2.1. *Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m$ be d-dimensional Gaussian random vectors of variance σ^2 centered at $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ respectively. Assuming $d \geq 3$ and $k \geq 3d$,*

$$\mathbf{Pr}\left[\exists i : \mathbf{dist}\left(\boldsymbol{a}_i, \boldsymbol{b}_i\right) \geq k\sigma\right] \leq m e^{-k^2/4}.$$

In particular, for $k = 8d\sqrt{\log(m/\sigma)}$, this probability is at most $m e^{-16d^2 \log(m/\sigma)}$.

We will make use of this lemma throughout the paper, and we will fix the value of k throughout the paper to

$$k = 8d\sqrt{\log(m/\sigma)}.$$

We will frequently assume that $\mathbf{dist}\left(\boldsymbol{a}_i, \boldsymbol{b}_i\right) \leq k\sigma$. As the probability that this assumption is false is at most $m e^{-16d^2 \log(m/\sigma)}$, and $P_{\boldsymbol{b}}$ has at most $\binom{m}{d}$ vertices, the estimate of the expected number of steps taken by the simplex algorithm that we make under this assumption can be off by at most an additive

$$m e^{-16d^2 \log(m/\sigma)} \binom{m}{d} < 1.$$

2.1 Well-scaled polytopes

In this section, we show that if the polytope defined by (5) is well-scaled and contains the origin, then it is close to the polytope defined by (6). Note that the origin is inside the polytope if and only if $y_i = 1$ for all i. The results in this section are not necessary for the results in the rest of the paper.

For every polytope in d dimensions, there is a linear transformation under which the polytope contains a ball of radius 1 and is contained in a ball of radius d (*c. f.* [12, 3.1.9]) with the same center. In this section, we assume that the polytope $P_{\boldsymbol{a}}$ is contained between two concentric balls, with the origin as the center, of radii 1 and R, respectively. In particular, we assume that, for all i, $1/R \leq \|\boldsymbol{a}_i\| \leq 1$.

Let $P_{\boldsymbol{a}}$ be the polytope defined by $\boldsymbol{a}_1, \ldots \boldsymbol{a}_m$ and $P_{\boldsymbol{b}}$ be the polytope defined by $\boldsymbol{b}_1, \ldots \boldsymbol{b}_m$. We will now show that $P_{\boldsymbol{a}}$ and $P_{\boldsymbol{b}}$ do not differ too much.

LEMMA 2.2. *Let $\boldsymbol{a}_1, \ldots \boldsymbol{a}_m$ be planes that determine a polytope $P_{\boldsymbol{a}}$ that contains the ball of radius 1 around the origin and is contained in the ball of radius R around the origin. Let $\boldsymbol{g}_1, \ldots \boldsymbol{g}_m$ be Gaussian random vectors of variance σ^2 centered at the origin. Let $\boldsymbol{b}_i = \boldsymbol{a}_i + \boldsymbol{g}_i$ and let $P_{\boldsymbol{b}}$ be the polytope given by planes $\boldsymbol{b}_1, \ldots \boldsymbol{b}_m$. Then for $\sigma < \epsilon/Rk$, where we recall $k = 8d\sqrt{\log(m/\sigma)}$, the following hold with probability at least $1 - m e^{-16d^2 \log(m/\sigma)}$.*

(a) For every direction, given by a unit vector $\boldsymbol{z}$,

$$\begin{aligned} (1-\epsilon)\mathbf{opt}_{\boldsymbol{z}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) &\leq \mathbf{opt}_{\boldsymbol{z}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m) \\ &\leq \frac{\mathbf{opt}_{\boldsymbol{z}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)}{1-\epsilon}, \end{aligned}$$

where by

$$\mathbf{opt}_{\boldsymbol{z}}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m),$$

we mean the solution to (5) assuming $y_i = 1$ for each i.

(b) *For every vertex* $\boldsymbol{v}$ *of* $P_{\boldsymbol{b}}$ *and every linear function* $\boldsymbol{z}$ *optimized by* $\boldsymbol{v}$*, the angle of* $\boldsymbol{v}$ *to* $\boldsymbol{z}$ *is at most* $\arccos(1-\epsilon/(R+\epsilon))$.

3. THE MAIN ARGUMENT

Our main result is that for any two vectors $\boldsymbol{t}$ and $\boldsymbol{z}$, the expected size of

$$\mathbf{optSimps}_{\boldsymbol{z},\boldsymbol{t};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$$

is polynomial in m, d, and $1/\sigma$.

THEOREM 3.1. *Let* $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_m$ *be points of norm at most* 1 *and let* $\boldsymbol{z}$ *and* $\boldsymbol{t}$ *be unit vectors. Let* $\mu_1,\ldots,\mu_m$ *be Gaussian measures of variance* $\sigma^2 < 1$ *centered at* $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_m$ *respectively, and let* $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m$ *be random points distributed independently according to* $\mu_1,\ldots,\mu_m$*. Then, the expected size of*

$$\mathbf{optSimps}_{\boldsymbol{z},\boldsymbol{t};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$$

is at most $f(d,m,\sigma)+2$*, where*

$$f(d,m,\sigma) = 2\pi\frac{2\cdot 10^{12}(d+1)d^4m^2(1+8d\sqrt{\log(m/\sigma)}\sigma)^{11}}{\sigma^{12}}.$$

PROOF. To bound the number of simplices in this set, let $\boldsymbol{q}$ be a unit vector in the span of $\boldsymbol{t}$ and $\boldsymbol{z}$ that is perpendicular to $\boldsymbol{z}$. Then, let

$$\boldsymbol{z}_\theta = \boldsymbol{z}\cos(\theta)+\boldsymbol{q}\sin(\theta).$$

We want to bound the size of

$$\mathbf{E}_{\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m}\left[\left|\left\{\bigcup_\theta \mathbf{optSimp}_{\boldsymbol{z}_\theta;\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right\}\right|\right] \quad (7)$$

Our approach is to consider N points located uniformly around the circle, $\theta_0,\ldots,\theta_{N-1}$, and to compute

$$\mathbf{E}_{\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m}\left[\left|\left\{\bigcup_{i\in[0,\ldots,N-1]} \mathbf{optSimp}_{\boldsymbol{z}_{\theta_i};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right\}\right|\right] \quad (8)$$

for N sufficiently large that it is exceedingly unlikely that any simplices have been missed.

We will bound (8) by counting those i for which $\boldsymbol{z}_{\theta_i}$ and $\boldsymbol{z}_{\theta_{i+1}}$ have different optimal simplicies. That is, we use

$$\begin{aligned}&\left|\left\{\bigcup_{i\in[0,\ldots,N-1]} \mathbf{optSimp}_{\boldsymbol{z}_{\theta_i};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right\}\right| \\ &= \left|\left\{i : \mathbf{optSimp}_{\boldsymbol{z}_{\theta_i};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m) \neq \mathbf{optSimp}_{\boldsymbol{z}_{\theta_{((i+1) \mod N)}};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right\}\right|\end{aligned}$$

We let

$$\begin{aligned}\delta &= \sigma/(4\sqrt{e}m^{2d}d^2), \text{ and}\\ N &= 7(2(1+k\sigma)/\delta)^d.\end{aligned}$$

The angle between $\boldsymbol{z}_{\theta_i}$ and $\boldsymbol{z}_{\theta_{i+1}}$ is $2\pi/N$. Applying Theorem 3.3 with $\epsilon = 2\pi/N$, we find that if $\mathbf{optSimp}_{\boldsymbol{z}_{\theta_i};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$ is non-empty, then the chance that this simplex contains a cone radius ϵ around $\boldsymbol{z}_{\theta_i}$, and therefore $\boldsymbol{z}_{\theta_{i+1}}$, is at least

$$1 - (2\pi/N)\left(\frac{2\cdot 10^{12}(d+1)d^4m^2(1+8d\sqrt{\log(m/\sigma)}\sigma)^{11}}{\sigma^{12}}\right) - me^{-11d^2\log(m/\sigma)}.$$

Therefore, the probability that $\boldsymbol{z}_{\theta_i}$ and $\boldsymbol{z}_{\theta_{i+1}}$ are optimized by different simplices is at most

$$(2\pi/N)\left(\frac{2\cdot 10^{12}(d+1)d^4m^2(1+8d\sqrt{\log(m/\sigma)}\sigma)^{11}}{\sigma^{12}}\right) + me^{-11d^2\log(m/\sigma)}.$$

So, the expected number of simplices that optimize rays in $\{z_{\theta_0},\ldots,z_{\theta_{N-1}}\}$ is at most

$$f(m,d,\sigma) + Nme^{-11d^2\log(m/\sigma)}.$$

Assuming $m \geq d \geq 3$, one can show $Nme^{-11d^2\log(m/\sigma)} \leq 1$. So, the number of simplices that optimize rays in $z_{\theta_0},\ldots,z_{\theta_{N-1}}$ is at most

$$f(m,d,\sigma)+1.$$

To bound the probability that these N points have missed a simplex entirely, we use a result whose proof appears in the full paper which implies that the probability that the intersection of any simplex with $\mathbf{Span}(\boldsymbol{t},\boldsymbol{z})$ subtends an angle less than

$$\left(\frac{\delta}{2(1+k\sigma)}\right)^d \geq 2\pi/N$$

is at most

$$\binom{m}{d}\binom{d}{2}\left(4\sqrt{e}(\delta/\sigma)\right) + me^{-16d^2\log(m/\sigma)} \leq \frac{1}{\binom{m}{d}}.$$

As this bad event occurs with probability less than the total number of possible simplices, it can add at most 1 to the expected size of

$$\mathbf{optSimps}_{\boldsymbol{z},\boldsymbol{t};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m).$$

□

3.1 Integral Formulation

Let $\boldsymbol{z}$ be a unit vector and let $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m$ be the perturbations of $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_m$. In this section, we will show that it is unlikely that $\boldsymbol{z}$ makes a small angle with any vector in the boundary of $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$. That is, there is probably a cone around $\boldsymbol{z}$ of reasonably large angle that is strictly contained in the cone from the origin through $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$. The proof has two parts: we show that the point of intersection of the ray through $\boldsymbol{z}$ with $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)$ is probably far from the boundary of the simplex, and we show that it is unlikely that $\boldsymbol{z}$ intersects the simplex at a small angle. To make this formal, we define

DEFINITION 3.2. *For a unit vector* $\boldsymbol{z}$ *and vectors* $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d$ *for which the ray from the origin through* $\boldsymbol{z}$ *intersects the simplex spanned by* $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d$*,*

$$\mathbf{ang}(\boldsymbol{z},\partial\mathbf{Simplex}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d))$$

is the minimum of the angle of $\boldsymbol{z}$ *to* $\boldsymbol{v}$*, where* $\boldsymbol{v}$ *ranges over the boundary of* $\mathbf{Simplex}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)$.

Let $p^{(1)}_{\boldsymbol{z}}(\epsilon)$ be the probability that $\boldsymbol{z}$ is ϵ close in angle to the boundary of the simplex it intersects:

$$p^{(1)}_{\boldsymbol{z}}(\epsilon) \stackrel{\text{def}}{=} \mathbf{Pr}_{\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m}\left[\mathbf{ang}\left(\boldsymbol{z}, \partial\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right) < \epsilon\right].$$

So that this makes sense when the polytope is unbounded in direction $\boldsymbol{z}$ or infeasible, we adopt the convention that the angle of $\boldsymbol{z}$ to the empty set is 0.

The main result of this section is

THEOREM 3.3. *Let $\boldsymbol{z}$ be a unit vector and let $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_m$ be points of norm at most 1. Let $\mu_1,\ldots,\mu_m$ be Gaussian measures of variance σ^2 centered at $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_m$ respectively, and let $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m$ be random points distributed independently according to $\mu_1,\ldots,\mu_m$. Then,*

$$p^{(1)}_{\boldsymbol{z}}(\epsilon) \leq \epsilon\frac{2\cdot 10^{12}(d+1)d^4m^2(1+k\sigma)^{11}}{\sigma^{12}} + me^{-11d^2\log(m/\sigma)}.$$

To prove Theorem 3.3, we analyze an integral expression for $p^{(1)}_{\boldsymbol{z}}(\epsilon)$. To simplify the notation in this integral, let $\Pi(d,m)$ be the set of all permutations on m elements in which the first d elements are sorted and the last $m-d$ elements are sorted. For example,

$$\Pi(2,4) = \{(1,2,3,4),(1,3,2,4),(1,4,2,3),(2,3,1,4),(2,4,1,3),(3,4,1,2)\}.$$

The other notational convention we introduce is

DEFINITION 3.4. *For an event A, we let*

$$[A]$$

denote the function that is 1 if A is true and 0 otherwise.

Let $\mathbf{Cone}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)$ denote the cone from the origin through $\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)$.

DEFINITION 3.5. *For a permutation π, let*

$$\boldsymbol{CHy}\left(\pi(1),\ldots,\pi(d)|\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m\right)$$

be the event that there exists $\boldsymbol{\omega}\in S^d$ and r such that

$$\begin{aligned}\left\langle\boldsymbol{\omega}|\boldsymbol{b}_{\pi(i)}\right\rangle &= y_i r, \text{ for } 1\leq i\leq d, \text{ and}\\ \left\langle\boldsymbol{\omega}|\boldsymbol{b}_{\pi(i)}\right\rangle &\leq y_i r, \text{ for } 1\leq i\leq m.\end{aligned}$$

In the case that each $y_i = 1$, this is equivalent to the condition that $\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)$ be a facet of the convex hull of $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m$.

By Claim 1.1, we have

$$\begin{aligned}\left[\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right) = \mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m)\right]&\\ = \left[\mathbf{CH}\boldsymbol{y}\left(\pi(1),\ldots,\pi(d)|\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m\right)\right]\cdot&\\ \left[\boldsymbol{z}\in\mathbf{Cone}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)\right]&\end{aligned}$$

That is, $(\boldsymbol{\omega}, 1/r)$ is the polar coordinate representation of $\boldsymbol{x}$ from Claim 1.1.

Using this notation, we write:

LEMMA 3.6.

$$\begin{aligned}p^{(1)}_{\boldsymbol{z}}(\epsilon) =&\\ \sum_{\pi\in\Pi(d,m)}\int_{\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m}&\left[\boldsymbol{CHy}\left(\pi(1),\ldots,\pi(d)|\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m\right)\right]\cdot\\ &\left[\boldsymbol{z}\in\mathbf{Cone}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)\right]\cdot\\ &\left[\mathbf{ang}\left(t,\partial\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}\right)\right)<\epsilon\right]\cdot\\ &\mu_1(\boldsymbol{b}_1)\cdots\mu_n(\boldsymbol{b}_n)\,d\boldsymbol{b}_1,\ldots,d\boldsymbol{b}_m.\end{aligned}$$

PROOF. We have enumerated over every possible choice of simplex, and set up the integral so that it is zero if that choice of simplex is not the one that optimizes $\boldsymbol{z}$. □

3.2 Focusing on one simplex

Our analysis will be by brute force: we will bound the integral for every choice of π. We apply more brute force by introducing a change of variables that allows us to restrict $\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}$ to an arbitrary plane. In the new variables, we specify the plane in which $\boldsymbol{b}_{\pi(1)},\ldots,\boldsymbol{b}_{\pi(d)}$ lie using polar coordinates, and we then specify their locations using local coordinates in that plane. The Jacobian of this change is computed by a famous theorem of integral geometry due to Blaschke [4] (for more modern treatments, see [21] or [23, 12.24]).

THEOREM 3.7 (BLASCHKE). *For $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d$ variables taking values in $\mathbb{R}^d$, $\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d$ taking values in $\mathbb{R}^{d-1}$, $\boldsymbol{\omega}\in S^{d-1}$, $r\in\mathbb{R}$, let*

$$P:(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)\mapsto((\boldsymbol{\omega},r),\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d),$$

denote the map from the d points $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d$ to the plane through those points represented in spherical coordinates as $(\boldsymbol{\omega},r)$, and the locations of those points on the plane, $\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d$. The Jacobian of the map P is

$$||P|| = \mathbf{Vol}\left(\mathbf{Simplex}\left(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\right).$$

That is,

$$d\boldsymbol{b}_1\ldots d\boldsymbol{b}_d = \mathbf{Vol}\left(\mathbf{Simplex}\left(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\right)d\boldsymbol{\omega}\,dr\,d\boldsymbol{c}_1\ldots d\boldsymbol{c}_d$$

We let $\nu^{\boldsymbol{\omega},r}_{\pi(i)}(\boldsymbol{c}_i)$ denote the restriction of $\mu_{\pi(i)}$ to the plane $(\boldsymbol{\omega},r)$. The restriction of a Gaussian distribution to a plane is a Gaussian distribution with the same variance centered at the projection of the center of the original to the plane; so, $\nu^{\boldsymbol{\omega},r}_{\pi(i)}$ has variance σ^2 and is centered at $\hat{\boldsymbol{a}}_i$, the projection of $\boldsymbol{a}_i$ to the plane $(\boldsymbol{\omega},r)$. To help express $p^{(1)}_{\boldsymbol{z}}(\epsilon)$ in these new variables, we let

$$\mathbf{Cone}\left(\boldsymbol{\omega},r,\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\stackrel{\text{def}}{=}\mathbf{Cone}\left(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d\right).$$

LEMMA 3.8.

$$\begin{aligned}p^{(1)}_{\boldsymbol{z}}(\epsilon) =&\\ \sum_{\pi\in\Pi(d,m)}\int_{\boldsymbol{\omega},r}\prod_{i=d+1}^{m}&\left(\int_{\boldsymbol{b}_{\pi(i)}}\left[\left\langle\boldsymbol{\omega}|\boldsymbol{b}_{\pi(i)}\right\rangle\leq y_i r\right]\mu_{\pi(i)}(\boldsymbol{b}_{\pi(i)})\,d\boldsymbol{b}_{\pi(i)}\right)\cdot\\ &\int_{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d}\left[\boldsymbol{z}\in\mathbf{Cone}\left(\boldsymbol{\omega},r,\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\right]\\ &\left[\mathbf{ang}\left(\boldsymbol{z},\partial\mathbf{Simplex}\left(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\right)<\epsilon\right]\cdot\\ &\prod_{i=1}^{d}\nu^{\boldsymbol{\omega},r}_{\pi(i)}(\boldsymbol{c}_i)\mathbf{Vol}\left(\mathbf{Simplex}\left(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d\right)\right)\\ &d\boldsymbol{\omega}\,dr\,d\boldsymbol{c}_1\cdots d\boldsymbol{c}_d\end{aligned}$$

From this expression, one can see that the distribution on $\boldsymbol{b}_{\pi(d+1)}, \ldots, \boldsymbol{b}_{\pi(m)}$ can influence the distribution of $(\boldsymbol{\omega}, r)$, but once $\boldsymbol{\omega}$ and r are fixed, they do not effect the distribution of the points on that plane.

3.3 Division into Distance and Angle

Our bound on $\mathbf{ang}\left(\boldsymbol{z}, \partial\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right)$ will have two parts. We first let $\boldsymbol{p}^{\boldsymbol{\omega},r}$ denote the intersection of plane $(\boldsymbol{\omega}, r)$ with the ray through $\boldsymbol{z}$, and we bound

$$\mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right)\right),$$

which denotes the distance from $\boldsymbol{p}^{\boldsymbol{\omega},r}$ to the boundary of the simplex. We then bound the angle between $\boldsymbol{z}$ and $\boldsymbol{\omega}$. We combine these bounds with the following lemma:

LEMMA 3.9. *Let $\boldsymbol{z}$ be a unit vector defining a ray that intersects $\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right)$ and let $\boldsymbol{p}^{\boldsymbol{\omega},r}$ be the point of intersection. Let*

$$h = \mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right)\right).$$

Then,

$$\begin{aligned}\mathbf{ang}&\left(\boldsymbol{z}, \partial\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right)\right)\\ &\geq \arctan(h\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle / (\left\|\boldsymbol{p}^{\boldsymbol{\omega},r}\right\| + h)).\end{aligned}$$

Using this lemma, we can bound $p_{\boldsymbol{z}}^{(1)}$ using the following lemma.

LEMMA 3.10. *For $\epsilon < 1/100$,*

$$p_{\boldsymbol{z}}^{(1)}(\epsilon) \leq p_{\boldsymbol{z}}^{(2)}(\epsilon) + me^{-16d^2\log(m/\sigma)},$$

where

$$p_{\boldsymbol{z}}^{(2)}(\epsilon) = \sum_{\pi\in\Pi(d,m)} \int_{\boldsymbol{\omega},r} K\left(\pi, \boldsymbol{\omega}, r, \frac{\epsilon}{\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle}\frac{2(1+k\sigma)}{.99}\right) d\boldsymbol{\omega}\; dr\,.$$

and

$$\begin{aligned}&K(\pi, \boldsymbol{\omega}, r, \epsilon) =\\ &\quad\prod_{i=d+1}^{m}\left(\int_{\boldsymbol{b}_{\pi(i)}} \left[\left\langle\boldsymbol{\omega}|\boldsymbol{b}_{\pi(i)}\right\rangle \leq y_i r\right] \mu_{\pi(i)}(\boldsymbol{b}_{\pi(i)})\, d\boldsymbol{b}_{\pi(i)}\right)\cdot\\ &\quad\int_{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d} [\boldsymbol{z}\in\mathbf{Cone}\left(\boldsymbol{\omega}, r, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)]\cdot\\ &\quad\left[\mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right) < \epsilon\right]\cdot\\ &\quad\prod_{i=1}^{d}\nu_{\pi(i)}^{\boldsymbol{\omega},r}(\boldsymbol{c}_i)\mathbf{Vol}\left(\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right)\; d\boldsymbol{c}_1\cdots d\boldsymbol{c}_d\,.\end{aligned}$$

So, $p_{\boldsymbol{z}}^{(2)}(\epsilon)$ is roughly the probability that

$$.99\cdot\mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right)\right)\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle / 2(1+k\sigma)$$

is less than ϵ.

PROOF. Follows by Lemma 3.9,

$$1/100 > \arctan(x) \quad\Longrightarrow\quad \arctan(x) > (99/100)x,$$

and the fact that the probability that

$$\begin{aligned}\left\|\boldsymbol{p}^{\boldsymbol{\omega},r}\right\| &\geq 1+k\sigma, \text{ or}\\ \mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right) &\geq 1+k\sigma\end{aligned}$$

is at most $me^{-16d^2\log(m/\sigma)}$. □

3.4 Bounding the distance

Roughly speaking, we want to bound

$$\mathbf{Pr}_{\boldsymbol{b}_1,\ldots,\boldsymbol{b}_m}\left[\mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)\right) < \epsilon\right].$$

In particular, for every choice of π, $\boldsymbol{\omega}$, and r, we bound

$$K(\pi, \boldsymbol{\omega}, r, \epsilon)/K(\pi, \boldsymbol{\omega}, r, \infty).$$

Once we fix π, $\boldsymbol{\omega}$, and r, the term

$$\prod_{i=d+1}^{m}\left(\int_{\boldsymbol{b}_{\pi(i)}} \left[\left\langle\boldsymbol{\omega}|\boldsymbol{b}_{\pi(i)}\right\rangle < y_i r\right] \mu_{\pi(i)}(\boldsymbol{b}_{\pi(i)})\, d\boldsymbol{b}_{\pi(i)}\right)$$

becomes a constant. Using the observation that

$$[\boldsymbol{z}\in\mathbf{Cone}\left(\boldsymbol{\omega}, r, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)] \Leftrightarrow \left[\boldsymbol{p}^{\boldsymbol{\omega},r}\in\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right],$$

we see that the integral:

$$\begin{aligned}\int_{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_d}\quad &\left[\boldsymbol{p}^{\boldsymbol{\omega},r}\in\mathbf{Simplex}\left(\boldsymbol{c}_0, \ldots, \boldsymbol{c}_d\right)\right]\\ &\left[\mathbf{dist}\left(\boldsymbol{p}^{\boldsymbol{\omega},r}, \partial\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right) < \epsilon\right]\cdot\\ &\mathbf{Vol}\left(\mathbf{Simplex}\left(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_d\right)\right)\prod_{i=1}^{d}\nu_{\pi(i)}^{\boldsymbol{\omega},r}(\boldsymbol{c}_i),\end{aligned}$$

is proportional to $K(\pi, \boldsymbol{\omega}, r, \epsilon)$. A theorem whose proof occupies a substantial fraction of the full paper implies

$$\begin{aligned}\frac{K(\pi, \boldsymbol{\omega}, r, \epsilon)}{K(\pi, \boldsymbol{\omega}, r, \infty)} &\leq (d+1)\epsilon\left(\frac{97d^2(1+k\sigma)^3}{\sigma^4}\right)^2\\ &+ me^{-16d^2\log(m/\sigma)}.\end{aligned} \tag{9}$$

3.5 Bounding the Angle

Let $\boldsymbol{\omega}$ be the unit normal to $\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)$. We show that it is unlikely that $\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle$ is small. In particular, we show that the probability that $\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle$ is less than ϵ is at most proportional to ϵ^2. Our argument uses quite a bit of brute force: we show that this is true under the assumption that

$$\mathbf{optSimp}_{\boldsymbol{z};\boldsymbol{y}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m) = \mathbf{Simplex}\left(\boldsymbol{b}_{\pi(1)}, \ldots, \boldsymbol{b}_{\pi(d)}\right),$$

for *any* π.

For any π, we can express the probability that $\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle < \epsilon$ by

$$\frac{\int_{\boldsymbol{\omega}\in S^{d-1}:0\leq\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle<\epsilon}\int_{r>0} K(\pi, \boldsymbol{\omega}, r, \infty)\, d\boldsymbol{\omega}\; dr}{\int_{\boldsymbol{\omega}\in S^{d-1}}\int_{r>0} K(\pi, \boldsymbol{\omega}, r, \infty)\, d\boldsymbol{\omega}\; dr} \tag{10}$$

To simplify this expression, let

$$J(\pi, \boldsymbol{\omega}, r) \stackrel{\text{def}}{=} K(\pi, \boldsymbol{\omega}, r, \infty).$$

In the full paper we prove that for every choice of π

$$\begin{aligned}\frac{\int_{\boldsymbol{\omega}\in S^{d-1}:0\leq\left\langle\boldsymbol{\omega}|\boldsymbol{z}\right\rangle<\epsilon}\int_r J(\pi, \boldsymbol{\omega}, r)\, d\boldsymbol{\omega}\; dr}{\int_{\boldsymbol{\omega}\in S^{d-1}}\int_r J(\pi, \boldsymbol{\omega}, r)\, d\boldsymbol{\omega}\; dr} &<\\ \epsilon^2(2\cdot 10^7)m^2(1+k\sigma)^4/\sigma^4 + me^{-16d^2\log(m/\sigma)}.\end{aligned} \tag{11}$$

3.6 Combining the bounds

To combine the bounds on angle and distance, we apply the following lemma:

LEMMA 3.11. *Let $L(\boldsymbol{\omega}, \epsilon)$ be a function such that*

$$\frac{L(\boldsymbol{\omega}, \epsilon)}{L(\boldsymbol{\omega}, \infty)} \leq \min\left(1, c_2\epsilon + c_0\right),$$

and

$$\frac{\int_{\boldsymbol{\omega}:0\le\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle\le\epsilon} L(\boldsymbol{\omega},\infty)\,d\boldsymbol{\omega}}{\int_{\boldsymbol{\omega}:0\le\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle} L(\boldsymbol{\omega},\infty)\,d\boldsymbol{\omega}} \le c_1\epsilon^2 + c_0.$$

Then

$$\frac{\int_{\boldsymbol{\omega}:0\le\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle} L(\boldsymbol{\omega},\epsilon/\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle)\,d\boldsymbol{\omega}}{\int_{\boldsymbol{\omega}:0\le\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle} L(\boldsymbol{\omega},\infty)\,d\boldsymbol{\omega}} \le (4c_2+\epsilon)c_1\epsilon + (4c_2+2)c_0.$$

Proof of Theorem 3.3 We first bound $p^{(2)}_{\boldsymbol{z}}(\epsilon)$. Equation (11) says that for all π,

$$\frac{\int_{\boldsymbol{\omega}\in S^{d-1}:0\le\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle<\epsilon}\int_r K(\pi,\boldsymbol{\omega},r,\infty)\,d\boldsymbol{\omega}\;dr}{\int_{\boldsymbol{\omega}\in S^{d-1}}\int_r K(\pi,\boldsymbol{\omega},r,\infty)\,d\boldsymbol{\omega}\;dr} < \epsilon^2 c_1 + c_0,$$

where

$$\begin{aligned} c_1 &= (2\cdot 10^7)m^2(1+k\sigma)^4/\sigma^4, \text{ and} \\ c_0 &= me^{-16d^2\log(m/\sigma)}. \end{aligned}$$

Moreover, equation (9) says that for all π, $\boldsymbol{\omega}$ and r

$$\frac{K(\pi,\boldsymbol{\omega},r,\epsilon)}{K(\pi,\boldsymbol{\omega},r,\infty)} < \epsilon c_2 + c_0,$$

where

$$c_2 = (d+1)\left(\frac{97d^2(1+k\sigma)^3}{\sigma^4}\right)^2\epsilon.$$

So, we can apply Lemma 3.9 to show that for all π

$$\frac{\int_{\boldsymbol{\omega},r} K\left(\pi,\boldsymbol{\omega},r,\frac{\epsilon}{\langle\boldsymbol{\omega}|\boldsymbol{z}\rangle}\frac{2(1+k\sigma)}{.99}\right)d\boldsymbol{\omega}\;dr\,,}{\int_{\boldsymbol{\omega},r} K(\pi,\boldsymbol{\omega},r,\infty)\,d\boldsymbol{\omega}\;dr\,.} \le$$

$$\epsilon(4c_2+(2\epsilon(1+k\sigma)/.99))c_1((1+k\sigma)/.99)+(4c_2+2)c_0.$$

Summing over π, we get

$$p^{(2)}_{\boldsymbol{z}}(\epsilon) \le \epsilon(4c_2+(2\epsilon(1+k\sigma)/.99))c_1((1+k\sigma)/.99)+(4c_2+2)c_0.$$

Applying Lemma 3.10 and simplifying, we then obtain for $\epsilon < 1/100$ and $m-1 \ge d \ge 3$,

$$p^{(1)}_{\boldsymbol{z}}(\epsilon) \le \frac{2\cdot 10^{12}(d+1)d^4m^2(1+k\sigma)^{11}}{\sigma^{12}} + me^{-11d^2\log(m/\sigma)}.$$

□

Of course, if the original polytope were contained between two spheres as described in 2.1, we could apply Lemma 2.2 and the angle analysis would be unnecessary and much tighter bounds could be derived.

4. PHASE I

Phase I of a two-phase simplex algorithm has the job of finding an initial vertex on the polytope determined by the constraints of a linear program, or determining that the constraints are infeasible. It is a relatively simple task to find a vertex of a polytope that contains the origin. If the origin is not feasible, then Phase I algorithms typically construct a linear program in one higher dimension whose solution provides a vertex of the original polytope, if such a vertex exists. Ideally, the higher-dimension polytope should have an easily identifiable vertex. In this section, we describe a Phase I simplex algorithm that we will employ to find a vertex of the constraint polytope whether or not it contains the origin. We will again use the shadow-vertex pivot rule. To prepare for Phase II, this algorithm will also choose a vector $\boldsymbol{t}$ optimized by the vertex it finds.

Consider a polytope $P_{\boldsymbol{b}}$ given by the equations

$$\langle\boldsymbol{x}|\boldsymbol{b}_i\rangle \le y_i, \text{ for } 1\le i\le m.$$

where $y_i \in \{1,-1\}$ and $\boldsymbol{b}_i \in \mathbb{R}^d$. We construct a polytope $P_{\bar{\boldsymbol{b}}}$ in $d+1$ dimensions that has $P_{\boldsymbol{b}}$ as a facet, and for which we can easily find a vertex.

If we think of $\mathbb{R}^d$ as being coordinatized by $\boldsymbol{x}=(x_1,\ldots,x_d)$, then we appeand an extra dimension by prepending a variable x_0 and coordinatizing $\mathbb{R}^{d+1}$ by $\bar{\boldsymbol{x}}=(x_0,\ldots,x_d)$. We extend the constraints to handle x_0 by chosing m Gaussian random variables $g_{0,1},\ldots,g_{0,m}$ of variance σ^2 and mean 0, and setting

$$\bar{\boldsymbol{b}}_i \stackrel{\text{def}}{=} \begin{cases} [g_i,\boldsymbol{b}_i] & \text{if } i\le d, \text{ and} \\ [-\Psi+g_i,\boldsymbol{b}_i] & \text{if } i>d, \end{cases} \tag{12}$$

for some sufficiently large Ψ satisfying conditions explained later. We also add two constraints given by

$$\langle\bar{\boldsymbol{b}}_0|\bar{\boldsymbol{x}}\rangle \le y_0, \text{ and } \langle\bar{\boldsymbol{b}}_{-1}|\bar{\boldsymbol{x}}\rangle \le y_{-1},$$

where

$$\begin{aligned} \bar{\boldsymbol{b}}_0 &\stackrel{\text{def}}{=} [-1,0,\ldots,0], \text{ and } y_0 \stackrel{\text{def}}{=} 0, \\ \bar{\boldsymbol{b}}_{-1} &\stackrel{\text{def}}{=} [1,0,\ldots,0], \text{ and } y_{-1} \stackrel{\text{def}}{=} 1. \end{aligned} \tag{13}$$

Together, these imply $0\le x_0\le 1$. Moreover, $P_{\boldsymbol{b}}$ is obtained by restricting $P_{\bar{\boldsymbol{b}}}$ to $x_0=0$.

If Ψ satisfies certain conditions related to the least singular value of the matrix of $\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d$, which we denote $\kappa(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)$, then it easy to find a vertex of $P_{\bar{\boldsymbol{b}}}$:

LEMMA 4.1. *Let $\boldsymbol{v}$ be the solution to*

$$\langle\boldsymbol{v}|\boldsymbol{b}_i\rangle = y_i, \text{ for } 1\le i\le d,$$

and let Ψ satisfy

$$\Psi \ge k_1\sigma + \frac{\sqrt{d}k_1\sigma\,(\max_i\|\boldsymbol{b}_i\|)}{\kappa(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)} + \max_{d+1\le i\le m}\langle\boldsymbol{v}|\boldsymbol{b}_i\rangle - y_i,$$

where

$$k_1 \stackrel{\text{def}}{=} 4\sqrt{d\log m}.$$

Let $\bar{\boldsymbol{b}}_{-1},\ldots,\bar{\boldsymbol{b}}_m$ and $y_{-1},\ldots,y_m$ be as defined in this section. Let $\bar{\boldsymbol{v}}$ be the solution to

$$\langle\bar{\boldsymbol{v}}|\bar{\boldsymbol{b}}_i\rangle = y_i, \text{ for } i\in\{-1,1,\ldots,d\}.$$

Then, with probability at least $1-me^{-4d\log m}$ over the choice of $g_{0,1},\ldots,g_{0,m}$, $\bar{\boldsymbol{v}}$ is a vertex of $P_{\bar{\boldsymbol{b}}}$. For each direction $\boldsymbol{t}$ optimized by $\boldsymbol{v}$, the vector

$$\bar{\boldsymbol{t}} \stackrel{\text{def}}{=} \left[\frac{\sqrt{d}\,(\max_i\|\boldsymbol{b}_i\|)}{\kappa(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)},\boldsymbol{t}\right] \tag{14}$$

is optimized by $\bar{\boldsymbol{v}}$.

We now define our two-phase shadow vertex algorithm. Our Phase I minimizes x_0 over $P_{\bar{\boldsymbol{b}}}$ using the shadow-vertex simplex algorithm. The starting vertex will be $\bar{\boldsymbol{v}}$ as defined in Lemma 4.1, and the start vector $\bar{\boldsymbol{t}}$ will be as described in (14) where $\boldsymbol{t}$ is chosen from a subset of $\textbf{Simplex}(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)$ by choosing $\beta_1,\ldots,\beta_d$ uniformly at random subject to $\sum_i\beta_i = 1$ and $\beta_i \ge 1/3d^2$, and setting $\boldsymbol{t}=\sum\beta_i\boldsymbol{b}_i$.

Let $\bar{\boldsymbol{w}}$ be the first vertex such that $w_0 = 0$ encountered by the shadow-vertex simplex algorithm as it minimizes x_0 starting from $\bar{\boldsymbol{t}}$. $\bar{\boldsymbol{w}}$ is the vertex of $P_{\boldsymbol{b}}$ that optimizes $\boldsymbol{t}$. We run the phase II shadow vertex algorithm on $P_{\boldsymbol{b}}$ starting from vertex $\bar{\boldsymbol{w}}$ and initial direction $\boldsymbol{t}$.

THEOREM 4.2. *Let $\boldsymbol{z}$ be a unit vector and let $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ be points of norm at most 1. Let $\mu_1, \ldots, \mu_m$ be Gaussian measures of variance $\sigma^2 < 1$ centered at $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ respectively, and let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m$ be random points distributed independently according to $\mu_1, \ldots, \mu_m$. Then, for any $\eta < 1$, with probability at least $1 - 2\eta$, the number of steps taken by the two-phase shadow-vertex simplex algorithm as it solves the linear program (6) is at most*

$$\left(5\left(f(d+1, m, \sigma_2/(1+\Psi)) + f(d, m, \sigma) + 6\right) + 2\right)/\eta,$$

where

$$\Psi = \frac{42\sqrt{e}(1 + 4\sqrt{d \log m}\sigma_2)^2 d^2}{10\eta\sigma},$$

and

$$\sigma_2 = \frac{\eta^2\sigma^2}{5600 d^6 \log m},$$

and where $f(d, m, \sigma)$ is as defined in Theorem 3.1.

The proof has two steps. We first imagine that $P_{\boldsymbol{a}}$ is a polytope in which $\mathbf{Simplex}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_d)$ has good aspect ratio. We then let $P_{\boldsymbol{b}}$ be a slight Gaussian perturbation of $P_{\boldsymbol{a}}$ as usual. Given that $\mathbf{Simplex}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_d)$ has good aspect ratio and the perturbation is small, the initial vector we choose, $\boldsymbol{t}$, will intersect $\mathbf{Simplex}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_d)$ and be chosen almost uniformly from those well-inside this simplex. By showing that most choices of vectors $\boldsymbol{t}$ intersecting well-inside $\mathbf{Simplex}(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_d)$ result in both the Phase I and Phase II algorithms running quickly, we can show that our choice of vector $\boldsymbol{t}$ in $\mathbf{Simplex}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ probably results in the Phase I and Phase II algorithms running quickly. We then prove the theorem by considering any $P_{\boldsymbol{a}}$ and decomposing the perturbation $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m$ into two perturbations: one after which the first simplex almost definitely has good aspect ratio, and a smaller perturbation that we can analyze by the preceeding analysis.

5. DISCUSSION AND OPEN QUESTIONS

The results proved in this paper support the assertion that the shadow-vertex simplex algorithm usually runs in polynomial time. However, our understanding of the performance of the simplex algorithm is far from complete. In this section, we discuss problems in the analysis of the simplex algorithm and in the smoothed analysis of algorithms that deserve further study.

5.1 Practicality of the analysis

While we have demonstrated that the smoothed complexity of the shadow-vertex algorithm is polynomial, the polynomial we obtain is quite large. Yet, we believe that the present analysis provides some intuition for why the shadow-vertex simplex algorithm should run quickly. It is clear that the proofs in this paper are very loose and make many worst-case assumptions that are unlikely to be simultaneously valid. In the full version of the paper, we give some indication as to how one should be able to reduce the σ^{12} term in Theorem 3.1 to σ^4.

5.2 Further analysis of the simplex algorithm

- While we have analyzed the shadow-vertex pivot rule, there are many other pivot rules that are more commonly used in practice. Knowing that one pivot rule usually takes a polynomial walk makes it seem reasonable that others should as well. We consider the maximum-increase and steepest-increase rules to be good candidates for smoothed analysis.

- As many sparse linear programming problems arise in practice, the *relative smoothed complexity* of the shadow vertex simplex method should be analyzed.

- Many linear programs encountered in practice have degeneracies. Is there a model of perturbations that result in or preserve degeneracies? We note that some simplex methods handle the problem of degeneracies by slightly perturbing the linear program.

- Even if we cannot perform a smoothed anlysis of other pivot rules, we might be able to measure the diameter of a polytope under smoothed analysis. It would be interesting to know if it is expected to be polynomial in m, d, and $1/\sigma$.

- Given that the shadow-vertex simplex algorithm can solve the perturbations of linear programs efficiently, it seems natural to ask if we can follow the solutions as we *unperturb* the linear programs. For example, having solved an instance of type (6), it makes sense to follow the solution as we let σ approach zero. Such an approach is often called a *homotopy* or *path-following* method. So far, we know of no reason that there should exist an A for which one cannot follow these solutions in expected polynomial time, where the expectation is taken over the choice of G. Of course, if one could follow these solutions in expected polynomial time for every A, then one would have a randomized strongly-polynomial time algorithm for linear programming!

5.3 Smoothed Analysis

We believe that many algorithms will be better understood through smoothed analysis. Scientists and engineers routinely use algorithms with poor worst-case performance. Often, they solve problems that appear intractable from the worst-case perspective. While we do not expect smoothed analysis to explain every such instance, we hope that it can explain away a significant fragment of the discrepancy between the algorithmic intuitions of engineers and theorists.

To aid the reader in the application of smoothed analysis, we now outline a generalization of the definitions made in Section 1.2. The key is to generalize the notion of perturbation.

Let X be a space of inputs containing X_n for all n. For each $x \in X$, we define a monotone increasing neighborhood system around x parameterized by a real variable δ taking values between zero and infinity such that

1. $S_0^x = \{x\}$.
2. For $\delta < \delta'$, we have $S_\delta^x \subseteq S_{\delta'}^x$. (monotone increasing)

We then define a family of decaying density functions on these neighborhood systems parameterized by a real variable σ taking values between zero and infinity that should satisfy

1. μ_σ^x is a probability measure.

2. For all δ, σ, $y \in S_\delta^x$ and $z \notin S_\delta^x$, we have $\mu_\sigma^x(y) > \mu_\sigma^x(z)$. (decaying)

3. For all δ and $\sigma < \sigma'$, $\mu_\sigma^x(S_\delta^x) > \mu_{\sigma'}^x(S_\delta^x)$. (concentration)

4. For all δ, $\lim_{\sigma \to 0} \mu_\sigma^x(S_\delta^x) \to 1$. (concentration in limit)

With these measures defined, we can generalize the definition of smoothed complexity of an algorithm A on inputs of length n to

$$\max_{x \in X_n} \mathbf{E}_{y \overset{\mu_\sigma^x}{\leftarrow} X} [C(A, y)]$$

The definition of ϵ-smoothed complexity can be similarly generalized.

As these definitions are very broad, one must be careful to exercise good taste when applying them. In particular, one should be very cautious when applying them to the analysis of any discrete combinatorial algorithm. We expect that the appropriate notion of perturbation for algorithms with discrete inputs such as graphs will depend heavily on the algorithm being analyzed. A good indication of whether a smoothed analysis is meaningful is whether it allows approximation for a general collection of input instances.

6. ACKNOWLEDGEMENTS

We thank Chloe for support in writing this paper. We thank Marcos Kiwi and Steve Vavasis for comments on earlier drafts of this paper. We thank Alan Edelman, Charles Leiserson, and Michael Sipser for helpful conversations. Finally, we thank Alan Edelman for proposing the name "smoothed analysis".

7. REFERENCES

[1] I. Adler, R. M. Karp, and R. Shamir. A simplex variant solving an m x d linear program in $O(min(m^2, d^2))$ expected number of pivot steps. *J. Complexity*, 3:372–387, 1987.

[2] I. Adler and N. Megiddo. A simplex algorithm whose average number of steps is bounded between two quadratic functions of the smaller dimension. *Journal of the ACM*, 32(4):871–895, Oct. 1985.

[3] D. Avis and V. Chvátal. Notes on Bland's pivoting rule. In *Polyhedral Combinatorics*, volume 8 of *Math. Programming Study*, pages 24–34. 1978.

[4] W. Blaschke. Integralgeometrie 2: Zu ergebnissen von m. w. crofton. *Bull. Math. Soc. Roumaine Sci.*, 37:3–11, 1935.

[5] K. H. Borgwardt. *Untersuchungen zur Asymptotik der mittleren Schrittzahl von Simplexverfahren in der linearen Optimierung*. PhD thesis, Universitat Kaiserslautern, 1977.

[6] K. H. Borgwardt. *The Simplex Method: a probabilistic analysis*. Number 1 in Algorithms and Combinatorics. Springer-Verlag, 1980.

[7] G. B. Dantzig. Maximization of linear function of variables subject to linear inequalities. In T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*, pages 339–347. 1951.

[8] A. Edelman. Eigenvalue roulette and random test matrices. In M. S. Moonen, G. H. Golub, and B. L. R. D. Moor, editors, *Linear Algebra for Large Scale and Real-Time Applications*, NATO ASI Series, pages 365–368. 1992.

[9] S. Gass and T. Saaty. The computational algorithm for the parametric objective function. *Naval Research Logistics Quarterly*, 2:39–45, 1955.

[10] D. Goldfarb. Worst case complexity of the shadow vertex simplex algorithm. Technical report, Columbia University, 1983.

[11] D. Goldfarb and W. T. Sit. Worst case behaviour of the steepest edge simplex method. *Discrete Applied Math*, 1:277–285, 1979.

[12] M. Grotschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1991.

[13] R. G. Jeroslow. The simplex algorithm with the pivot rule of maximizing improvement criterion. *Discrete Math.*, 4:367–377, 1973.

[14] G. Kalai. A subexponential randomized simplex algorithm. In *Proc. 24th Ann. ACM Symp. on Theory of Computing*, pages 475–482, Victoria, B.C., Canada, May 1992.

[15] G. Kalai and D. J. Kleitman. A quasi-polynomial bound for the diameter of graphs of polyhedra. *Bulletin Amer. Math. Soc.*, 26:315–316, 1992.

[16] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[17] L. G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademia Nauk SSSR*, pages 1093–1096, 1979.

[18] V. Klee and G. J. Minty. How good is the simplex algorithm ? In Shisha, O., editor, *Inequalities – III*, pages 159–175. Academic Press, 1972.

[19] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, Oct./Nov. 1996.

[20] N. Megiddo. Improved asymptotic analysis of the average number of steps performed by the self-dual simplex algorithm. *Mathematical Programming*, 35(2):140–172, 1986.

[21] R. E. Miles. Isotropic random simplices. *Adv. Appl. Prob.*, 3:353–382, 1971.

[22] K. G. Murty. Computational complexity of parametric linear programming. *Math. Programming*, 19:213–219, 1980.

[23] L. A. Santalo. *Integral Geometry and Geometric Probability*. Encyclopedia of Mathematics and its Applications. Addison-Wesley, 1976.

[24] S. Smale. The problem of the average speed of the simplex method. In *Proceedings of the 11th International Symposium on Mathematical Programming*, pages 530–539, August 1982.

[25] S. Smale. On the average number of steps in the simplex method of linear programming. *Mathematical Programming*, 27:241–262, 1983.

[26] M. Todd. Polynomial expected behavior of a pivoting algorithm for linear complementarity and linear programming problems. *Mathematical Programming*, 35:173–192, 1986.