Bahman Kalantari

# Linear Programming, Convex Hull Dualities, Diagonal Matrix Scaling, Applications and Extensions

October 8, 2017

Springer

# Contents

# Chapter 1
# The Triangle Algorithm and Its Analysis

**Abstract** In this chapter we introduce the *Triangle Algorithm*, a simple geometric algorithm for solving the convex hull membership problem. The algorithm makes use of the *distance duality* proved in a previous chapter. It either finds a point in the convex hull of a finite set of points that is as close to a given point as desired, or it computes a hyperplane that separates the specific point from the convex hull, thereby proving the point lies outside of the convex hull. In the latter case it actually computes a point inside the convex hull whose Voronoi region relative to the specific point contains the convex hull. In particular, it approximates the distance of the specific point to the convex hull, to within a factor of two. We analyze the complexity of the Triangle Algorithm and prove different complexity bounds.

## 1.1 Introduction

Given a set $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$ and a distinguished point $p \in \mathbb{R}^m$, we consider the problem of testing if $p$ lies in $conv(S)$, the convex hull of $S$. We shall refer to this problem as the *convex hull membership problem*. The convex hull membership problem is a basic problem in computational geometry and a very special case of the *convex hull problem*, see Goodman and O'Rourke [22].

The convex hull membership problem is not only a fundamental problem in computational geometry, but in linear programming (LP). This can be argued on different grounds. On the one hand the problem is a very special case of LP. On the other hand, by the LP duality theory, the general LP problem may be cast as a single LP feasibility problem. The LP feasibility problem can then be converted into a convex hull membership problem via several different approaches. To argue the significance of the convex hull membership in linear programming, it can be justified that the two most famous polynomial-time LP algorithms, the ellipsoid algorithm of Khachiyan [36] and the projective algorithm of Karmarkar [34], are in fact explicitly or implicitly designed to solve a case of the convex hull membership problem where $p = 0$, see [24]. Furthermore, using an approach suggested by Chvátal, in [24] it is shown

that there is a direct connection between a general LP feasibility and this homogeneous case of the convex hull membership problem, over integer or real input data. For integer inputs all known polynomial-time LP algorithms - when applied to solve the convex hull membership problem - would exhibit a theoretical complexity that is polynomial in $m$, $n$, and the size of encoding of the input data, often denoted by $L$, see e.g. [36]. The number $L$ can generally be taken to be dependent on the logarithm of $mn$ and of the logarithm of the absolute value of the largest entry in the input data. These are sometimes known as *weakly polynomial-time* algorithms. No *strongly polynomial-time* algorithm is known for LP, i.e. an algorithm that would in particular solve the convex hull membership in time complexity polynomial in $m$ and $n$.

The convex hull membership problem finds applications in computational geometry itself, in particular among the variations of the convex hull problem. One such a variation is the *irredundancy problem*, the problem of computing all the vertices of $conv(S)$, see [22]. Clearly, any algorithm for LP can be used to solve the irredundancy problem by solving a sequence of $O(n)$ convex hull membership problems. Clarkson [9], has given a more efficient algorithm than the straightforward approach of solving $n$ linear programs. The complexity of his algorithm depends on the number of vertices of the convex hull. Furthermore, by using an approach originally due to Matoušek [43] for solving closely related linear programming problems, Chan [5] has given a faster algorithm for the irredundancy problem. What is generally considered to be the convex hull problem is a problem more complicated than the convex hull membership problem and the irredundancy problem, requiring the description of $conv(S)$ in terms of its vertices, facets and adjacencies, see Chazelle [6] who offers an optimal algorithm.

The convex hull membership problem can also be formulated as the minimization of a convex quadratic function over a simplex. This particular convex program has found applications in statistics, approximation theory, and machine learning, see e.g Clarkson [10] and Zhang [55] who consider the analysis of a greedy algorithm for the more general problem of minimizing certain convex functions over a simplex (equivalently, maximizing concave functions over a simplex). The oldest version of such greedy algorithms is Frank-Wolfe algorithm, [18]. Special cases of the problem include support vector machines (SVM), approximating functions as convex combinations of other functions, see e.g. Clarkson [10]. The problem of computing the closest point of the convex hull of a set of points to the origin, known as *polytope distance* is related to the case of convex hull membership problem where $p$ is the origin. In some applications the polytope distance refers to the distance between two convex hulls. Gilbert's algorithm [21] for the polytope distance problem is one of the earliest known algorithms for the problem. Gärtner and Jaggi [19] show Gilbert's algorithm coincides with Frank-Wolfe algorithm when applied to the minimization of a convex quadratic function over a simplex. In this case the algorithm is known as *sparse greedy approximation*. For many other results regarding the applications of the minimization of a quadratic function over a simplex, see the bibliographies in [55], [10] and [19]. Clarkson [10] gives a through analysis of Frank-Wolfe and

its variations. Another class of algorithms that are applicable to the convex hull problem are the so-called *first-order* methods, see Nesterov [47] and Lan et al [40].

Despite all previous approaches and achievements in solving the convex hull membership problem, various formulations and algorithms, investigation of this fundamental problem will most likely continue in the future. The chapter reveals new and interesting geometric properties of this fundamental convex hull problem, including characterization theorems. We have already seen a relevant duality proved in a previous chapter, the *distance duality* theorems. Here we utilize the distance dualities to describe the *Triangle Algorithm*. We analyze theoretical complexity of the Triangle Algorithm showing that it is very attractive for solving convex hull membership problem. Additionally, utilizing the Triangle Algorithm, together by proving a sensitivity theorem in a subsequent chapter, we will describe a new algorithm for the LP feasibility problem and analyze its complexity.

## 1.2 The Triangle Algorithm

Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, and $p$ a distinguished point in $\mathbb{R}^m$. In this section we describe a simple algorithm for testing if $p \in conv(S)$. We will use $d(u,v)$ for the Euclidean distance between $u, v \in \mathbb{R}^m$. We recall two definitions:

**Definition 1.1.** Given $p' \in conv(S)$, we say $v_j \in S$ is *p-pivot* (or just *pivot*) if

$$d(p', v_j) \geq d(p, v_j). \tag{1.1}$$

**Definition 1.2.** We say $p' \in conv(S)$ is a *p-witness* (or just *witness*) if

$$d(p', v_i) < d(p, v_i), \quad \forall v_i \in S. \tag{1.2}$$

We denote the set of all witnesses by $W_p$.

We shall refer to the algorithm to be described as the *Triangle Algorithm*. The justification in the name lies in the fact that in each iteration the algorithm searches for a triangle $\triangle pp'v_j$ where $v_j \in S$, $p' \in conv(S)$, such that $d(p', v_j) \geq d(p, v_j)$. Given that such triangle exists, it uses $v_j$ as a $p$-pivot to "pull" the current iterate $p'$ closer to $p$ to get a new iterate $p'' \in conv(S)$. If no such a triangle exists, then by the distance duality $p'$ is a $p$-witness certifying that $p$ is not in $conv(S)$. The steps of the algorithm are described in the box. Note that given the coordinates of $p'$ and $\alpha_i$'s that give its representation as a convex combination of the $v_i$'s, it takes $O(m)$ operations to compute $p''$ and $O(n)$ operations to compute the $\alpha_i'$'s.

**Triangle Algorithm** ($S = \{v_1, \ldots, v_n\}$, $p$, $\varepsilon \in (0,1)$)

- **Step 0.** (**Initialization**) Let $p' = v = \text{argmin}\{d(p, v_i) : v_i \in S\}$.
- **Step 1.** If $d(p,p') < \varepsilon d(p,v)$, output $p'$ as $\varepsilon$-approximate solution, stop. Otherwise, if there exists a pivot $v_j$, replace $v$ with $v_j$. If no pivot exists, then output $p'$ as a witness, stop.
- **Step 2.** Compute the *step-size*

$$\alpha = \frac{(p-p')^T(v_j - p')}{d^2(v_j, p')}. \tag{1.3}$$

Given the current iterate $p' = \sum_{i=1}^n \alpha_i v_i$, set the new *iterate* as:

$$p'' = (1-\alpha)p' + \alpha v_j = \sum_{i=1}^n \alpha_i' v_i,$$

$$\alpha_j' = (1-\alpha)\alpha_j + \alpha, \quad \alpha_i = (1-\alpha)\alpha_i, \quad \forall i \neq j. \tag{1.4}$$

Replace $p'$ with $p''$, $\alpha_i$ with $\alpha_i'$, for all $i = 1, \ldots, n$. Go to Step 1.

By an easy calculation that shift $p'$ to the origin, it follows that the point $p''$ in Step 2 is the closest point to $p$ on the line $p'v_j$. Since $p''$ is a convex combination of $p'$ and $v_j$ it will remain in $conv(S)$. The algorithm replaces $p'$ with $p''$ and repeats the above iterative step. Note that a $p$-pivot $v_j$ may or may not be a vertex of $conv(S)$. In the following we state some basic properties of the algorithm to be used in the analysis of its complexity.

## 1.3 Reduction of Gap and Its Worst-Case Analysis

In what follows we state a theorem that is fundamental in the analysis of the algorithm to be described in the next section. It relates to one iteration of the algorithm to test if $p$ lies in $conv(S)$ and reveals its worst-case performance.

In the theorem below the reader may consider $p'$ as a given point in $conv(S)$ and $v$ as a point in $S$ to be used as a *p-pivot* in order to compute a new point $p''$ in $conv(S)$ where the new gap $d(p'', p)$ is to be a reduction of the current gap $d(p', p)$.

**Theorem 1.1.** *Let $p, p', v$ be distinct points in $\mathbb{R}^m$. Suppose $d(p,v) \leq d(p',v)$. Let $p''$ be the point on the line segment $p'v$ that is closest to $p$. Assume $p'' \neq v$. Let $\delta = d(p',p)$, $\delta' = d(p'',p)$, and $r = d(p,v)$* (see Figure 1.1)*. Then,*

$$\delta' \leq \begin{cases} \delta\sqrt{1 - \frac{\delta^2}{4r^2}}, & \text{if } \delta \leq r; \\ r, & \text{otherwise.} \end{cases} \tag{1.5}$$

*Proof.* Given $\delta \leq r$, consider $p'$ as a variable $x'$ and the corresponding $p''$ as $x''$. We will consider the maximum value of $d(x'', p)$ subject to the desired constraints. We will prove

$$\delta^* = \max\left\{ d(x'',p): \quad x \in \mathbb{R}^m, \quad d(x',p) = \delta, \quad d(x',v) \geq r \right\} = \delta\sqrt{1 - \frac{\delta^2}{4r^2}}. \tag{1.6}$$

This optimization problem can be stated in the two-dimensional Euclidean plane. Assume $p \neq p''$, and consider the two-dimensional plane that passes through the points $p, p', v$. Given that $\delta \leq r$, $p'$ must lie inside or on the boundary of the circle of radius $\delta$ centered at $p$, but outside or on the boundary of the circle of radius $r$ centered at $v$, see Figure 1.1, circles $C$, $C'$, and $C''$.

Now consider the circle of radius $\delta$ centered at $p$, $C''$ in Figure 1.1. Consider the ratio $\delta'/r$ as $p'$ ranges over all the points on the circumference of $C''$ while outside or on the boundary of $C'$. It is geometrically obvious and easy to argue that this ratio is maximized when $p'$ is a point of intersection of the circles $C'$ and $C''$, denoted by $x^*$ in Figure 1.2. We now compute the corresponding ratio.

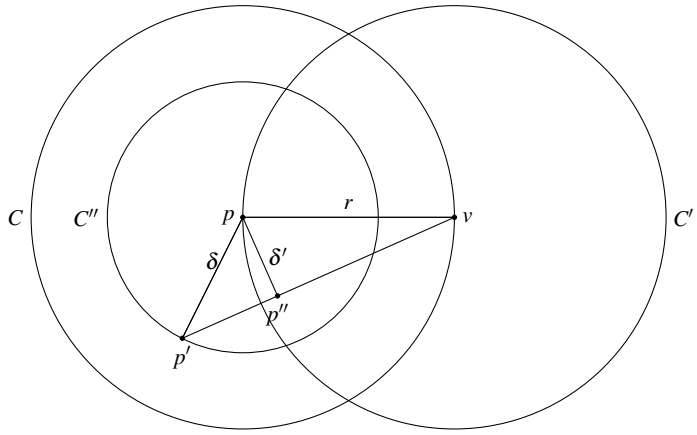


**Fig. 1.1** Depiction of gaps $\delta = d(p',p)$, $\delta' = d(p'',p)$, when $\delta \leq r = d(p,v)$.

Considering Figure 1.2, and the isosceles triangle $\triangle vpx^*$, let $h$ denote the length of the bisector line from $v$ to the base, and let $q$ denote the midpoint of $p$ and $x*$. Consider the right triangles $\triangle pvq$ and $\triangle px^*x^{**}$. The angle $\angle vpq$ is identical with $\angle px^*x^{**}$. Hence, the two triangles are similar and we may write

$$\frac{\delta^*}{\delta} = \frac{h}{r} = \frac{1}{r}\sqrt{r^2 - \frac{\delta^2}{4}} = \sqrt{1 - \frac{\delta^2}{4r^2}}. \tag{1.7}$$

This proves the first inequality in (1.5).

Next, suppose $\delta > r$. Figure 1.3 considers several possible scenarios for this case. If in the triangle $\triangle pvp'$ the angle $\angle pvp'$ is acute, the line segment $p'v$ must necessarily intersect $C'$. This implies $p''$ is the bisector of a chord in $C'$, hence inside

of $C'$. If the angle $\angle pvp'$ is at least $\pi/2$, then $p''$ will coincide with $v$. Hence, proving the second inequality in (1.5).

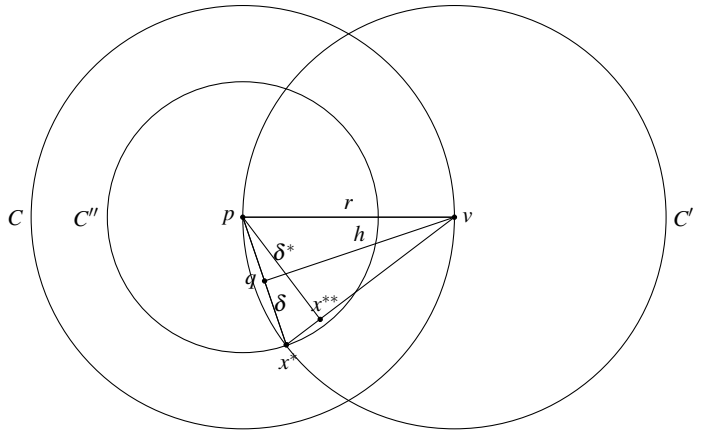


**Fig. 1.2** The worst-case scenario for the gap $\delta' = \delta^*$, when $\delta \leq r$.

## 1.4 The Complexity of Triangle Algorithm

We are now ready to analyze the complexity of the algorithm. Set

$$R = \max\left\{d(p,v_i), i = 1,\ldots,n\right\}. \tag{1.8}$$

**Corollary 1.1.** *Let $p, p', p'', v$ be as in Theorem 1.1, and $r = d(p,v)$, $\delta = d(p,p')$ , $\delta' = d(p,p'')$. If $p'' \neq v$ and $\delta \leq r$, then*

$$\delta' \leq \delta\sqrt{1-\frac{\delta^2}{4R^2}} < \delta\exp\left(-\frac{\delta^2}{8R^2}\right). \tag{1.9}$$

*Proof.* The first inequality follows from Theorem 1.1 and the definition of $R$. To prove the next inequality, we use that when $x \neq 0$, $1+x < \exp(x)$, and set $x = -\delta^2/4R^2$.

**Lemma 1.1.** *Assume $p$ is in conv$(S)$. Pick $p_0 \in$ conv$(S)$, let $\delta_0 = d(p_0,p)$. Assume $\delta_0 \leq R_0 = \min\{d(p,v_i), i = 1,\ldots,n\}$.*

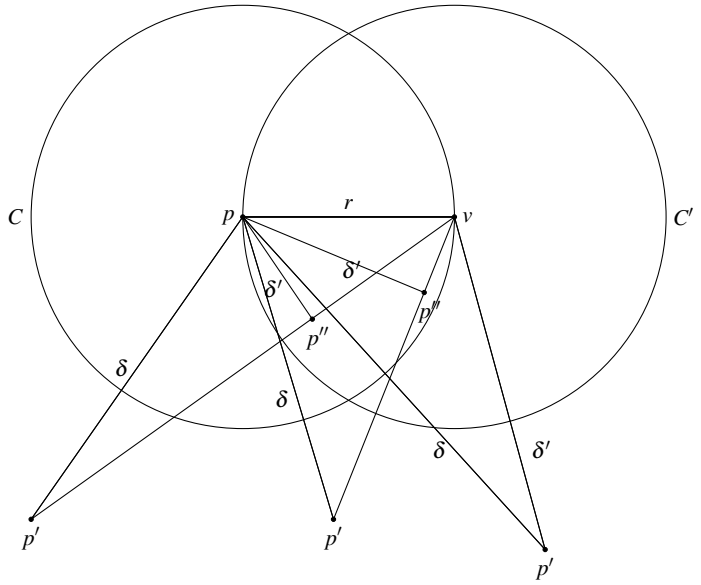


**Fig. 1.3** Depiction of gaps $\delta = d(p',p)$, $\delta' = d(p'',p)$, when $\delta > r = d(p,v)$.

*Let $k \equiv k(\delta_0)$ be the maximum number of iterations of the Triangle Algorithm in order to compute $p_k \in conv(S)$ so that if $\delta_j = d(p_j,p)$ for $j = 1,\ldots,k$, we have*

$$\delta_k < \frac{\delta_0}{2} \leq \delta_j, \quad j = 1,\ldots,k-1. \tag{1.10}$$

*Then, $k$ satisfies*

$$k = k(\delta_0) \leq \lceil N_0 \rceil, \quad N_0 \equiv N(\delta_0) = (32\ln 2)\frac{R^2}{\delta_0^2} < 23\frac{R^2}{\delta_0^2} \tag{1.11}$$

*Proof.* For each $j = 1,\ldots,k-1$, Corollary 1.1 applies. The repeated application of (1.9) in Corollary 1.1, the assumption (1.10) that for each such $j$, $\delta_j \geq \delta_0/2$, and the monotonicity of the exponential function implies

$$\delta_j < \delta_{j-1}\exp\left(-\frac{\delta_{j-1}^2}{8R^2}\right) \leq \delta_{j-1}\exp\left(-\frac{\delta_0^2}{32R^2}\right) < \delta_{j-2}\exp\left(-\frac{2\delta_0^2}{32R^2}\right) \leq \cdots \tag{1.12}$$

It follows that

$$\delta_{k-1} < \delta_0\exp\left(-\frac{(k-1)\delta_0^2}{32R^2}\right). \tag{1.13}$$

Thus from (1.9) and (1.13) we have

$$\delta_k < \delta_{k-1} \exp\left(-\frac{\delta_{k-1}^2}{8R^2}\right) \leq \delta_0 \exp\left(-\frac{k\delta_0^2}{32R^2}\right). \tag{1.14}$$

To have $\delta_k < \delta_0/2$, it suffices to satisfy

$$\exp\left(-\frac{k\delta_0^2}{32R^2}\right) \leq \frac{1}{2}. \tag{1.15}$$

Solving for $k$ in the above inequality implies the claimed bound in (1.11).

**Theorem 1.2.** *The Triangle Algorithm correctly solves the convex hull membership problem:*

*(i) Suppose $p \in conv(S)$. Given $\varepsilon > 0$, $p_0 \in conv(S)$, with $\delta_0 = d(p,p_0) \leq R_0 = \min\{d(p,v_i) : i = 1,\ldots,n\}$. The number of iterations $K_\varepsilon$ to compute a point $p_\varepsilon$ in $conv(S)$ so that $d(p,p_\varepsilon) < \varepsilon d(p,v_i)$, for some $v_i \in S$ satisfies*

$$K_\varepsilon \leq \frac{48}{\varepsilon^2} = O(\varepsilon^{-2}). \tag{1.16}$$

*(ii) Suppose $p \notin conv(S)$. If $\Delta$ denotes the distance from $p$ to $conv(S)$, i.e.*

$$\Delta = \min\left\{d(x,p): \quad x \in conv(S)\right\}, \tag{1.17}$$

*the number of iterations $K_\Delta$ to compute a $p$-witness, a point $p_\Delta$ in $conv(S)$ so that $d(p_\Delta,v_i) < d(p,v_i)$ for all $v_i \in S$, satisfies*

$$K_\Delta \leq \frac{48R^2}{\Delta^2} = O\left(\frac{R^2}{\Delta^2}\right). \tag{1.18}$$

*Proof.* From Lemma 1.1 and definition of $k(\delta_0)$ (see (1.11)), in order to half the initial gap from $\delta_0$ to $\delta_0/2$, in the worst-case the Triangle Algorithm requires $k(\delta_0)$ iterations. Then, in order to reduce the gap from $\delta_0/2$ to $\delta_0/4$ it requires at most $k(\delta_0/2)$ iterations, and so on. From (1.11), for each nonnegative integer $r$ the worst-case number of iterations to reduce a gap from $\delta_0/2^r$ to $\delta_0/2^{r+1}$ is given by

$$k\left(\frac{\delta_0}{2^r}\right) \leq \left\lceil N\left(\frac{\delta_0}{2^r}\right)\right\rceil = \lceil 2^{2r} N_0 \rceil \leq 2^{2r}\lceil N_0 \rceil. \tag{1.19}$$

Therefore, if $t$ is the smallest index such that $\delta_0/2^t < \varepsilon R$, i.e.

$$2^{t-1} \leq \frac{\delta_0}{R\varepsilon} < 2^t, \tag{1.20}$$

then the total number of iterations of the algorithm, $K_\varepsilon$, to test if condition (i) is valid satisfies:

$$K_\varepsilon \leq \lceil N_0 \rceil (1 + 2^2 + 2^4 + \cdots + 2^{2(t-1)}) \leq$$

$$\lceil N_0 \rceil \frac{2^{2t}-1}{3} \leq \lceil N_0 \rceil 2 \times 2^{2(t-1)} \leq (N_0+1)\frac{2\delta_0^2}{R^2\varepsilon^2} \tag{1.21}$$

From (1.11) we get

$$K_\varepsilon \leq \left(23\frac{R^2}{\delta_0^2}+1\right)\frac{2\delta_0^2}{R^2\varepsilon^2} = \left(23+\frac{\delta_0^2}{R^2}\right)\frac{2}{\varepsilon^2}. \tag{1.22}$$

Since $p \in conv(S)$ and from the definition of $R$ (see (1.8)) we have $\delta_0 = d(p,p_0) \leq R$, hence we get the claimed bound on $K_\varepsilon$ in (1.16).

Suppose $p \notin conv(S)$. If suffices to choose $\varepsilon = \Delta/R$. Then from (1.16) and the definition of $\Delta$ we get the bound on $K_\Delta$ in (1.18).

## 1.5 Strict Distance Duality

Here we first prove a stricter version of the distance duality, giving a stronger distance duality when $p \in conv(S)$. First we give a definition.

**Definition 1.3.** Given $p' \in conv(S)$, we say $v_j \in S$ is a *strict pivot* relative to $p$ (or *strict* $p$-pivot, or simply *strict* pivot) if $\angle p'pv_j \geq \pi/2$ (see Figure 1.4).

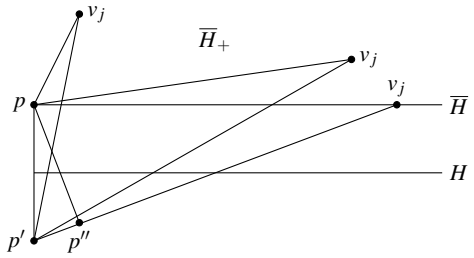


**Fig. 1.4** Several examples of strict $p$-pivot.

**Theorem 1.3. (Strict Distance Duality)** *Assume $p \notin S$. Then $p \in conv(S)$ if and only if for each $p' \in conv(S)$ there exists strict $p$-pivot, $v_j$. In particular,*

$$(p'-p)^T(v_j-p) \leq 0. \tag{1.23}$$

*Proof.* Suppose $p \in conv(S)$. Consider the orthogonal bisecting hyperplane to the line $p'p$, $H$, and the hyperplane parallel to it passing through $p$, $\overline{H}$, see Figure 1.4. Let $\overline{H}_+$ be the halfspace determined by this hyperplane that excludes $p'$. We claim it must contain a point $v_j \in S$ (see Figure 1.4). Otherwise, $p$ must be an extreme point

of $conv(S\cup\{p\})$, but since $p\notin S$, this implies $p\notin conv(S)$. This implies $\angle p'pv_j$ is at least $\pi/2$. Hence, (1.23) is satisfied.

Conversely, suppose that for each $p'\in conv(S)$, there exists $v_j\in conv(S)$ such that $\angle p'pv_j$ is at least $\pi/2$. If $p\notin conv(S)$, consider a witness $p'$. Then for each $v_j\in S$, $\angle p'pv_j<\pi/2$, a contradiction.

**Theorem 1.4.** *Given $p'\in conv(S)$, suppose $v_j$ is a strict p-pivot. Then if $\delta = d(p,p')$, $r=d(p,v_j)$, and $\delta' = d(p,p'')$, $p''$ the nearest to $p$ on the line segment $p'v_j$, we have*

$$\delta' = \frac{\delta r}{\sqrt{r^2+\delta^2}} \le \delta\sqrt{1-\frac{\delta^2}{2r^2}} \le \delta\exp\left(-\frac{\delta^2}{4r^2}\right). \tag{1.24}$$

*Proof.* It is easy to see that for a given strict pivot $v_j$, the worst-case of error occurs when $\angle p'pv_j=\pi/2$, (see Figure 1.4). Now using the similarity of the triangles $\triangle p'pv_j$ and $\triangle pp''p'$ (see Figure 1.4), we get the equality in the theorem. To prove the first inequality, we use the fact that for a positive number $x<1$,

$$\frac{1}{1+x}\le 1-\frac{x}{2}, \tag{1.25}$$

and let $x=\delta^2/r^2$. The second inequality follows from the inequality $1-x\le\exp(x)$.

Thus when $p\in conv(S)$ using a strict pivot we get a better constant in the worst-case complexity of the Triangle Algorithm than using any pivot.

**Definition 1.4.** We say $p'\in conv(S)$ is a *strict witness* relative to $p$ (or simply *strict* witness) if there is no strict pivot at $p'$. Equivalently, $p'$ is a strict witness if the orthogonal hyperplane to the line $p'p$ at $p$ separates $p$ from $conv(S)$. We denote the set of all strict witnesses by $\widehat{W}_p$.

Clearly $\widehat{W}_p$ contains $W_p$. However, interestingly while $W_p$ is not described by a set of linear inequalities, $\widehat{W}_p$ can be characterized by a set of strict linear inequalities. The following is straightforward.

**Proposition 1.1.** *We have*

$$\widehat{W}_p = \left\{x\in conv(S) : (x-p)^T(v_i-p)>0, i=1,\ldots,n\right\}.\Box \tag{1.26}$$

Clearly $p\in conv(S)$ if and only if $\widehat{W}_p$ is empty. Figure 1.5 shows the difference between $W_p$ and $\widehat{W}_p$. Its witness set was considered in an earlier example. The figure suggest that when $p\notin conv(S)$, using a strict pivot would detect the infeasibility of $p$ sooner than using any pivot.

**Theorem 1.5.** *Assume $p$ lies in the relative interior of $conv(S)$. Let $\rho$ be the supremum of radii of the balls centered at $p$ in this relative interior. Let $\delta_0 = d(p_0,p)$,*

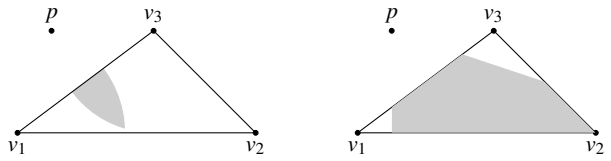


**Fig. 1.5** Witness set $W_p$ (left) and Strict Witness set $\widehat{W}_p$ (right).

$p_0 \in conv(S)$. *Let* $R = max\{d(p,v_i), i = 1,\ldots,n\}$. *Given* $\varepsilon \in (0,1)$, *suppose the Triangle Algorithm uses a strict pivot in each iteration. The number of arithmetic operations to compute* $p' \in conv(S)$ *such that* $d(p',p) < \varepsilon R$ *satisfies*

$$O\left(mn\frac{R^2}{\rho^2}\ln\frac{\delta_0}{\varepsilon R}\right), \quad \delta_0 = d(p_0,p). \tag{1.27}$$

*In particular, if* $\rho \geq \varepsilon^{1/t}R$, $t$ *a natural number then the complexity is*

$$O\left(mn\frac{1}{\sqrt[t]{\varepsilon^2}}\ln\frac{\delta_0}{\varepsilon R}\right). \tag{1.28}$$

*Proof.* Given the current iterate $p'$ where $d(p,p') \geq \varepsilon R$, let $q$ be the point on the extension of the line $p'p$, where $d(p,q) = \rho$, see Figure 1.6. Then we have $q \in conv(S)$. The orthogonal hyperplane to the line segment $pq$ must contain a point $v$ in $S$ on the side that excludes $p$, see Figure 1.6. For such strict pivot $v$, $\angle pqv$ is non-acute.

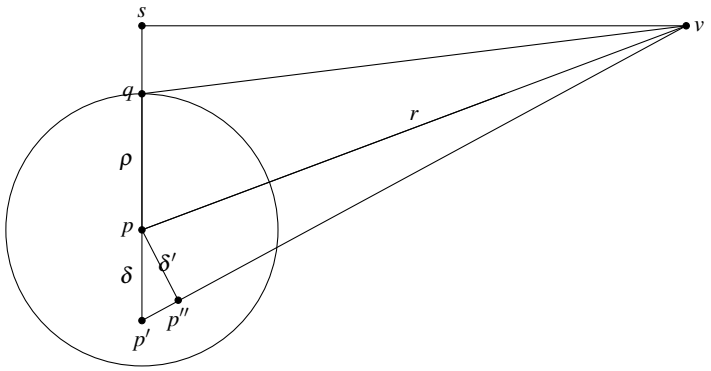


**Fig. 1.6** Existence of a strict pivot with reduction factor $1/\sqrt{1+\rho^2/R^2}$.

Let $p''$ be the next iterate. Then we have (see Figure 1.6):

$$\sin\angle pp'v = \frac{\delta'}{\delta}. \tag{1.29}$$

Let $r = d(p,v)$. We claim

$$\sin\angle pp'v \leq 1/\sqrt{1+\frac{\rho^2}{r^2}}. \tag{1.30}$$

To prove this, note that we have

$$\sin\angle pp'v \leq \sin\angle qpv. \tag{1.31}$$

However, considering that $\angle pqv$ is non-acute, we can introduce $s$ so that $\angle psv = \pi/2$. Thus we may write

$$\sin\angle qpv = \sin\angle spv = \frac{\sqrt{r^2-(d(q,s)+\rho)^2}}{r} \leq \frac{\sqrt{r^2-\rho^2}}{r} = \sqrt{1-\frac{\rho^2}{r^2}} \leq$$

$$1/\sqrt{1+\frac{\rho^2}{r^2}} \leq 1/\sqrt{1+\frac{\rho^2}{R^2}}. \tag{1.32}$$

Thus if $\delta_k$ represents the gap in $k$ iterations and $\delta_0$ the initial gap, then we have

$$\delta_k \leq \left(1/\sqrt{1+\frac{\rho^2}{R^2}}\right)^k \delta_0. \tag{1.33}$$

To have $\delta_k < \varepsilon R$ it suffices to find $k$ so that

$$-\frac{k}{2}\ln\left(1+\frac{\rho^2}{R^2}\right) < \ln\frac{\varepsilon R}{\delta_0}. \tag{1.34}$$

Equivalently,

$$\frac{k}{2}\ln\left(1+\frac{\rho^2}{R^2}\right) > \ln\frac{\delta_0}{\varepsilon R}. \tag{1.35}$$

As shown in previous theorem, for $u \in (0,1)$ we have, $\ln(1+u) \geq \frac{u}{2}$. Thus we choose $k$ satisfying

$$k > \frac{4R^2}{\rho^2}\ln\frac{\delta_0}{\varepsilon R}. \tag{1.36}$$

*Remark 1.1.* According to the above theorem good reduction factor occur when $p$ lies in a reasonable size ball in the relative interior of $conv(S)$. For instance, in the example of a square in Figure 1.7, aside from the case when $p$ is center of the square, for points in a reasonably large circle we would expect the complexity of the Triangle Algorithm to be very good. Only points near a boundary line can slow down the algorithm. However, even for these points when iteration begins to slow down

we can introduce *auxiliary pivots* to improve the reduction. This will be considered in a subsequent section.

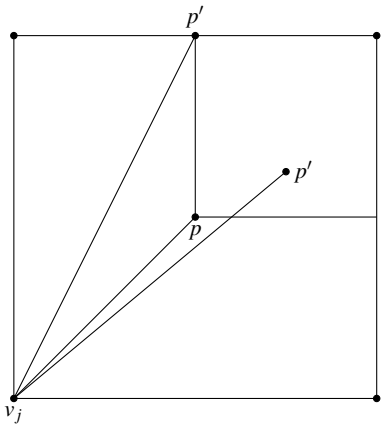


**Fig. 1.7** When $conv(S)$ is a square, and $p$ the center any pivot works well.

## 1.6 Problems

**1.1.** Suppose $p \in R^m$, and $S = \{v_1, v_2, \ldots, v_n\} \subset R^m$. If $p \in conv(S)$, assume it is known that the number of iterations of the *triangle algorithm* to get a point $p' \in conv(S)$ so that $d(p', p) \leq \varepsilon R$, $R = \max\{d(p, v_i) : i = 1, \ldots, n\}$, is bounded above by $48/\varepsilon^2$.

(i): Suppose $p \notin conv(S)$. In terms of distance of $p$ to $conv(S)$, i.e. $\Delta = \min\{d(p,x) : x \in conv(S)\}$, give a bound on the number of iterations to compute a *witness*.

(ii): Suppose $S = \{(0,0),(2,0),(0,2)\}$, $p = (2,1)$ (draw the triangle and $p$). What is the estimate of the number of iterations from part (i)? Can you bound the actual number of iteration no matter where you start in the $conv(S)$? Do the same with $p = (3,0)$.

(iii): What is the set of witnesses? What is the set of strict witnesses.

(iv): Can you estimate the number of iterations if $p = (1,1)$, say starting with $p' = (0,0)$?

## References


1. T. Asano, J. Matoušek, T. Tokuyama, Zone diagrams: Existence, uniqueness, and algorithmic challenge, *SIAM Journal on Computing*, 37 (2007), 1182 - 1198.
2. F. Aurenhammer, Voronoi Diagrams – A survey of fundamental geometric data structure, *ACM Computing Surveys*, 23 (1991), 345 - 405.
3. C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, 2 (1998), 121-167.
4. R. Berkowitz, B. Kalantari, D. Memendendez, and I. Kalantari, On properties of forbidden zone of polyhedrons and polytopes, Proceedings of the Ninth annual International Symposium on Voronoi Diagrams in Science and Engineering, (2012), 56 - 65.
5. T. M. Chan, Output-sensitive results on convex hulls, extreme points, and related problems, *Discrete Comput. Geom.*, 16 (1996), 369 - 387.
6. B. Chazelle, An optimal convex hull algorithm in any fixed dimension, Discrete Comput. Geom., 10 (1993), 377 - 409
7. V. Chvátal, *Linear Programming*, W.H. Freeman and Company, New York, 1983.
8. K.L. Clarkson, Las Vegas algorithms for linear and integer programming when the dimension is small, Proceedings of 29-th Annu. IEEE Sympos. Found. Comput. Sci., (1988), 452- 456.
9. K.L. Clarkson, More output-sensitive geometric algorithm, Proceedings of the 35-th Annual IEEE Symposium on Foundations of Computer Science, (1994) 695 - 702.
10. K. L. Clarkson, Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. In SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms, 922 - 931. Society for Industrial and Applied Mathematics, 2008.
11. G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.
12. G. B. Dantzig, An $\varepsilon$-precise feasible solution to a linear program with a convexity constraint in $1/\varepsilon^2$ iterations independent of problem size. Technical Report, Stanford University, 1992.
13. S. C. de Biasi, B. Kalantari, I. Kalantari, Maximal zone diagrams and their computation, Proceedings of the Seventh Annual International Symposium on Voronoi Diagrams in Science and Engineering (2010), 171 - 180.
14. S. C. de Biasi, B. Kalantari, I. Kalantari, Mollified zone diagrams and their computation, *Transactions on Computational Sciennce XIV: Special Issue on Voronoi Diagrams Delaunay Triangulation* (2011), 31 - 59.
15. R. Durier and C. Michelot, Sets of Efficient Points in a Normed Space, *Journal of Mathematical Analysis and Applications*, 117 (1986), 506 - 528.
16. M.E. Dyer and A.M. Frieze, A randomized algorithm for fixed-dimensional linear programming, *Math. Programming*, 44 (1989), 203 - 212.
17. M. Epelman and R.M. Freund, Condition number complexity of an elementary algorithm for resolving a conic linear system, Technical Report, 1997.
18. M. Frank and P. Wolfe, An algorithm for quadratic programming, *Naval Res. Logist. Quart.*, 3 (1956), 95 - 110.
19. B, Gärtner and M. Jaggi, Coresets for polytope distance, Symposium on Computational Geometry (2009), 33 - 42.
20. T. Gibson and B. Kalantari, Experiments with the triangle algorithm for linear systems, 2-page Extended Abstract, 23nd Annual Fall Workshop on Computational Geometry, City College of New York, 2013.
21. E. G. Gilbert, An iterative procedure for computing the minimum of a quadratic form on a convex set, *SIAM Journal on Control*, 4 (1966), 61 - 80.
22. J. E. Goodman, J. O'Rourke (Editors), *Handbook of Discrete and Computational Geometry*, 2nd Edition (Discrete Mathematics and Its Applications) 2004, Chapman & Hall Boca Raton.
23. S. Har-Peled, D. Roth, and D. Zimak, Maximum margin coresets for active and noise tolerant learning, IJCAI, 2007.

24. Y. Jin and B. Kalantari, A procedure of Chvátal for testing feasibility in linear programming and matrix scaling, *Linear Algebra and its Applications*, 416 (2006), 795 - 798.
25. G. Kalai, A subexponential randomized simplex algorithm, Proceedings of the 24-th Annual ACM Symposium on Theory of Computing, (1992), 475 - 482.
26. B. Kalantari, Canonical problems for quadratic programming and projective methods for their solution, *Contemporary Mathematics*, 114 (1990), 243 - 263.
27. B. Kalantari and M.R. Emamy-K, On linear programming and matrix scaling over the algebraic numbers, *Linear Algebra and its Applications*, 262 (1997), 283 - 306.
28. B. Kalantari, Semidefinite programming and matrix scaling over the semidefinite cone, *Linear Algebra and its Applications*, 375 (2003), 221 - 243.
29. B. Kalantari, A characterization theorem and an algorithm for a convex hull problem, arxiv.org/pdf/1204.1873v2.pdf, 2012.
30. B. Kalantari, Finding a lost treasure in convex hull of points from known distances. In the Proceedings of the 24th Canadian Conference on Computational Geometry (2012), 271 - 276.
31. B. Kalantari, Solving linear system of equations via a convex hull algorithm, arxiv.org/pdf/1210.7858v1.pdf, 2012.
32. B. Kalantari, An algorithmic separating hyperplane theorem and its applications, forthcoming.
33. B. Kalantari and M. Saks, On the triangle algorithm for the convex hull membership, 2-page Extended Abstract, 23nd Annual Fall Workshop on Computational Geometry, City College of New York, 2013.
34. N. Karmarkar, A new polynomial time algorithm for linear programming, *Combinatorica*, 4 (1984), 373 - 395.
35. J. A. Kelner and D. A. Spielman, A randomized polynomial-time simplex algorithm for linear programming, Proceedings of the 38th Annual ACM Symposium on Theory of Computing, 2006.
36. L. G. Khachiyan, A polynomial algorithm in linear programming, *Doklady Akademia Nauk SSSR*, (1979), 1093 - 1096.
37. L. Khachiyan and B. Kalantari, Diagonal matrix scaling and linear programming, *SIAM J. Optim.*, 4 (1992), 668 - 672.
38. V. Klee and G.J. Minty, How good is the simplex algorithm?, In O. Shisha, editor, Inequalities III, 159 - 175. Academic Press, 1972.
39. H.W. Kuhn, On a pair of dual nonlinear programs, in *Nonlinear Programming* (j. Abadie, Ed.), Wiley, New York, 1967, 37 - 54, see also, Locational problems and mathematical programming, in Mathematical Optimiation in Economics C.I.M.E. Summer Schools, Volume 38 (2011), 57 - 76.
40. G. Lan, Z. Lu, and R. D. C. Monteiro, Primal-dual first-order methods with $O(1/\varepsilon)$ iteration-complexity for cone programming, Math. Program., Ser. A 126 (2011), 1 - 29.
41. M. Li and B. Kalantari, Experimental study of the convex hull decision problem via a new geometric algorithm, 2-page Extended Abstract, 23nd Annual Fall Workshop on Computational Geometry, City College of New York, 2013.
42. J. Matoušek, M. Sharir, and E. Welzl, A subexponential bound for linear programming. In Proceedings of the 8th Annual ACM Symposium on Computational Geometry, (1992) 1 - 8.
43. J. Matoušek, Linear optimization queries, *Journal of Algorithms*, 14 (1993), 432 - 448.
44. N. Megiddo, Linear programming in linear time when the dimension is fixed, *Journal of the ACM* , 31 (1984), 114 - 127.
45. R. Motawani and R. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
46. Y.E. Nesterov, Smooth minimization of nonsmooth functions, Math. Program, 103 (2005), 127 - 152.
47. Y.E. Nesterov, private communications, 2013.
48. F. P. Preparata, M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.
49. R. T. Rockafellar, *Convex Analysis*, Princeton University Press Princeton, New Jersey 1997 [1970].

50. A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley and Sons, Inc., New York, Chichester, Brisbane, Toronto and Singapore, 1986.
51. R. Seidel, Small-dimensional linear programming and convex hulls made easy, Discrete Computational Geometry, 6 (1991), 423 - 434.
52. R. Shamir, The efficiency of the simplex method: A survey, Management Science, 33 (1987), 301 - 334.
53. M. Sharir and E. Welzl, A combinatorial bound for linear programming and related problems. In Proceedings of the 9th Symposium on Theoretical Aspects of Computer Science, (1992), 569 - 579.
54. M. J. Todd, The many facets of linear programming, Math. Prog., 91 (2002), 417 - 436.
55. T. Zhang, Sequential greedy approximation for certain convex optimization problems, IEEE Trans. Information Theory, 49 (2003), 682 - 691.
56. D. A. Zimak, Algorithms and Analysis for Multi-Category Classification, Ph.D. thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2006.