Finding a Lost Treasure in Convex Hull of Points From Known Distances

Bahman Kalantari*

Abstract

Given a set of points $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, and a set of positive numbers r_i, $i = 1, \ldots, n$, we wish to determine if there exists $p \in \text{Conv}(S)$ such that $d(p, v_i) = r_i$ for all $i = 1, \ldots, n$, where $d(\cdot, \cdot)$ denotes the Euclidean distance. We refer to this as the *ambiguous convex hull problem*. Given $\epsilon > 0$, we describe an algorithm that in $O(mn\epsilon^{-2} \ln \epsilon^{-1})$ arithmetic operations computes $p_\epsilon \in \text{Conv}(S)$ such that one of the three conditions hold; (1): $|d(p_\epsilon, v_i) - r_i| < \epsilon$, for all $i = 1, \ldots, n$; (2): $d(p_\epsilon, v_i) < r_i$, for all $i = 1, \ldots, n$; (3): $d(p_\epsilon, v_i) > r_i$, for all $i = 1, \ldots, n$. In case of (2), no point p with prescribed distances belongs to $\text{Conv}(S)$. In case of (3), no point p with prescribed distances exists. In case of (1), we give an estimate on $d(p_\epsilon, p)$. The algorithm is a variation of the Triangle Algorithm in [8] for the convex hull decision problem where p is given explicitly.

1 Introduction

The *convex hull decision problem* is to test if a given point $p \in \mathbb{R}^m$ lies in the convex hull of a given set of points $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, denoted by $\text{Conv}(S)$. This problem is a very special case of the *convex hull problem*, a problem that represents various descriptions of a polytope that is either specified as the convex hull of a finite point set or the intersection of a finite number of halfspaces, see Goodman and O'Rourke [5]. For more general convex hull problems and corresponding algorithms see, [5], Clarkson [4], Chan [1], Chazelle [2].

The convex hull decision problem is a fundamental problem in computational geometry and in linear programming (LP). A general LP problem can be formulated as a single LP feasibility problem, see e.g. Chvátal [3]. Then, given a bound on the norm of the vertices, a number that can be estimated for integer inputs, the latter problem in turn can be converted into the convex hull decision problem. Any LP algorithm can be applied to solve the convex hull decision problem. It can be argued that several polynomial-time algorithms for LP are in fact specifically designed to solve the convex hull decision problem with $p = 0$. These include, the ellipsoid algorithm of Khachiyan [11], the projective algorithm of Karmarkar[9], and the positive semidefinite diagonal matrix scaling algorithm of Khachiyan and Kalantari [12]. See [6] and also [7].

In a recent article, [8], we offered characterization theorems and a simple fully polynomial-time approximation algorithm, called the *Triangle Algorithm* for the convex hull decision problem having the following properties: Given $\epsilon \in (0, 1)$, in $O(mn\epsilon^{-2})$ arithmetic operations the algorithm computes a point $p_\epsilon \in \text{Conv}(S)$, where either $d(p_\epsilon, p) \leq \epsilon d(p, v_j)$ for some j; or for all $i = 1, \ldots, n$, $d(p_\epsilon, v_i) < d(p, v_i)$. The following characterization theorem in [8] plays an important role in the development and correctness of the Triangle Algorithm.

Theorem 1 *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$ be a given set of points and let $p \in \mathbb{R}^m$ be given. Then exclusively, either $p \in \text{Conv}(S)$ and for any $p' \in \text{Conv}(S)$ there exists $v_j \in S$ such that $d(p', v_j) \geq d(p, v_j)$, or $p \notin \text{Conv}(S)$ and there exists $p' \in \text{Conv}(S)$ such that $d(p', v_i) < d(p, v_i)$, for all $i = 1, \ldots, n$.*

Each $p' \in \text{Conv}(S)$ that satisfies $d(p', v_i) < d(p, v_i)$, for all $i = 1, \ldots, n$, acts as a *witness* to the infeasibility of p. The set W_p of all such witnesses is the intersection of $\text{Conv}(S)$ and open balls B_i of radius $d(p, v_i)$ centered at v_i, $i = 1, \ldots, n$ and forms a convex open set in the relative interior of $\text{Conv}(S)$. A corollary of Theorem 1 reveals a property of a set of intersecting open balls:

If a set of n open balls in $\mathbb{R}^m$ has a common boundary point p, their intersection is empty, if and only if p lies in the convex hull of their centers.

This property suggests we can define a geometric "dual" problem to the convex hull decision problem, the *intersecting open balls problem*:

Given a set of n open balls in $\mathbb{R}^m$ that are known to have a common boundary point p, determine if they have a nonempty intersection.

The Triangle Algorithm in particular is capable of solving the intersecting open balls problem. It can be used to solve some other versions of the convex hull problem, e.g. the *irredundancy problem* where all the vertices are to be determined. The Triangle Algorithm can also be used to solve a linear programming problem (see [8]) and as such offers an alternative to polynomial-time algorithms for linear programming, as well as the simplex method whose randomized version is shown to run in polynomial-time, see [10]. All known polynomial-time LP algorithms for integer inputs have operation complexity that is polynomial in m, n, and the size of

*Department of Computer Science, Rutgers University, kalantari@cs.rutgers.edu

encoding of the input data, often denoted by L, see [11]. As approximation schemes, polynomial-time algorithms for LP have complexity polynomial in the dimension of the data and $\ln(\epsilon^{-1})$. As is well-known, for integral inputs, by rounding, any approximate solution having sufficient precision can be turned into an exact solution. It is likely the Triangle Algorithm performs better than its worst-case analysis. Its average case or randomized versions could prove to give much better complexity. For fixed ϵ it produces approximate solutions in $O(mn)$ operations.

In this article we consider the *ambiguous convex hull problem*. It differs with the convex hull decision problem in that we do not know p explicitly, but we are given a set of distances r_i, $i = 1, \ldots, n$, presumed to equal $d(p, v_i)$. In particular, the distances may not even be valid distances of any point p in $\text{Conv}(S)$, or in its complement. Thus there is ambiguity to the problem. Two scenarios where the ambiguous convex hull problem applies well are as follows. The first, where we are given claimed distances of a hidden treasure from a set of known sites. The second, where we wish to determine the feasibility of building a site at prescribed distances in the convex hull of a given set of sites. Analogous to the Triangle Algorithm itself, the variation to be described here, called *Blindfold Triangle Algorithm*, is geometric in nature, simple in description, and very easy to implement, having worst-case complexity of $O(mn\epsilon^{-2} \ln \epsilon^{-1})$ arithmetic operations.

If a subset of $m + 1$ points in S that are in general position are identifiable, we can determine the coordinates of p by solving an $m \times m$ linear system, see Section 5. Doing so will reduce the problem to the convex hull decision problem. However, our goal is to avoid solving such linear systems. Furthermore, as will be seen in the final section of the article, we wish to argue that, at least in some cases, solving a linear system corresponds to an ambiguous convex hull problem. Thus such linear systems can be solved approximately in $O(m^2\epsilon^{-2} \ln(\epsilon^{-1}))$ (see Section 6).

While the complexity analysis of the Blindfold Triangle Algorithm is independent of Triangle Algorithm itself, it makes use of Theorem 1. For the sake of completeness, we will first give a proof of this theorem, somewhat different than the proof presented in [8].

2 A Voronoi Diagram-Based Proof of Theorem 1

In this section we present a simple proof of Theorem 1. Suppose $p \in \text{Conv}(S)$. Let p' be any point in $\text{Conv}(S) - \{p\}$. Consider $V(p) = \{x \in \mathbb{R}^m : \quad d(p, x) < d(p', x)\}$, i.e. the Voronoi cell of p with respect to the two point set $\{p, p'\}$. We wish to show $\overline{V}(p) = \{x \in \mathbb{R}^m : \quad d(p, x) \leq d(p', x)\}$ intersects S. If not, then S is a subset of $V(p') = \{x \in \mathbb{R}^m : \quad d(p', x) < d(p, x)\}$. Since $V(p')$ is convex, it follows that it must contain $\text{Conv}(S)$. This contradicts that $p \in \text{Conv}(S)$.

Conversely, suppose $p \notin \text{Conv}(S)$. Let p' be the point in $\text{Conv}(S)$ that is closest to p. We claim p' is a witness, i.e. $d(p', v_i) < d(p, v_i)$ for all i. For any $v_i \neq p'$, the angle $\angle pp'v_i$ cannot be acute since otherwise this would contradict p' being the closest point of $\text{Conv}(S)$ to p. This implies that $d(p', v_i) < d(p, v_i)$. If $p' = v_j$ for some j, then clearly the inequality is also satisfied for v_j. Hence the proof of the Theorem 1.

Remark This simple proof also implies the separating hyperplane theorem for the convex hull of a finite point set. The converse is however not true, that is, the separating hyperplane theorem does not imply Theorem 1. In this sense Theorem 1 is a stronger separation theorem. Theorem 1 can also be stated for a polytope that is described by the intersection of halfspaces. The proof however would have to rely on a further result: any point in the polytope can be written as the convex combination of its vertices.

3 Getting Closer to The Treasure

Given a point $p' \in \text{Conv}(S)$, as the current estimate to the location of p, we wish to compute a new point $p'' \in \text{Conv}(S)$ that reduces the current distance $d(p', p)$, referred here as the *gap*. However, since the location of p is unknown we must select p'' in such a way that guarantees improvement. The following theorem describes how this can be achieved.

Theorem 2 *Let $\epsilon > 0$ be given. Consider three given points $p, p', v \in \mathbb{R}^m$ satisfying the following conditions:*
(i) $r' = d(p', v) \geq r = d(p, v)$.
(ii) $\delta = d(p', p) \geq \epsilon$.
Let p'' be the point on the line segment $p'v$ such that

$$d(p', p'') = \frac{\epsilon^2}{2r'}. \tag{1}$$

Then if $d(p'', p) = \delta'$, we have

$$\frac{\delta'}{\delta} \leq \sqrt{1 - \frac{\epsilon^2}{2(r'^2 + r^2)}}. \tag{2}$$

In particular, if $r' \leq 2r$, then

$$\frac{\delta'}{\delta} \leq \sqrt{1 - \frac{\epsilon^2}{10r^2}}. \tag{3}$$

Proof. Without loss of generality we may assume that p, p', v lie in a Euclidean plane, where $v = (0, 0)$, $p' = (r', 0)$ and $p = (x, y)$, see Figure 1. Consider the concentric circles of radii r and r' centered at v. Let q be a point lying on the circle of radius r', satisfying $d(p', q) = \epsilon$. See Figure 1 where one of the two such

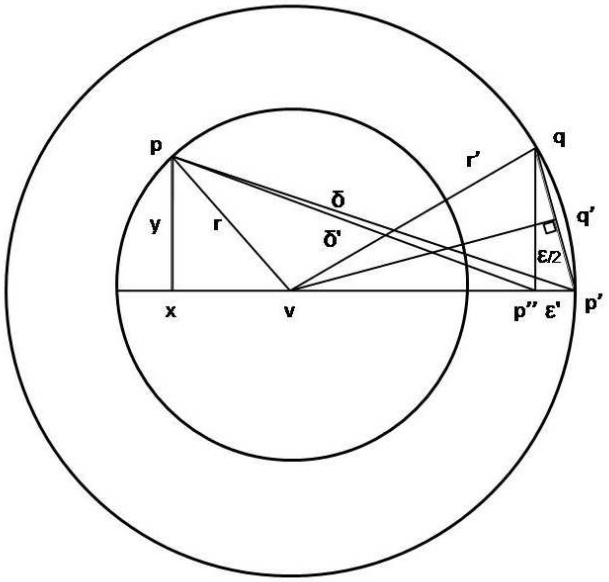

Figure 1: Reduction of $\delta = d(p', p)$ to $\delta' = d(p'', p)$.

points are shown. Draw the line from q perpendicular to the line $p'v$ and let p'' be the base of this line. We claim that this point coincides with p'' as defined in (1). We do so by computing $d(p', p'') = \epsilon'$. Let q' be the midpoint of the chord qp'. By a property of a circle, the line vq' is perpendicular to the line qp'. Consider the right triangles $\triangle qp'p''$ and $\triangle q'p'v$. They are similar since they have a common angle, $\angle qp'p''$. Therefore, we have

$$\frac{d(p', p'')}{d(p', q)} = \frac{d(p', q')}{d(v, p')}. \tag{4}$$

Substituting, equivalently we have,

$$\frac{\epsilon'}{\epsilon} = \frac{\epsilon/2}{r'}. \tag{5}$$

This gives (1). From this, that $p = (x, y)$, and since by assumption (ii) $\delta \geq \epsilon$, it follows that

$$x \in [-r, \rho], \quad \rho = \min\{r, r' - \epsilon'\}. \tag{6}$$

Now consider x as a variable ranging in the interval $[-r, \rho]$. Since $y^2 = r^2 - x^2$, the corresponding ratio, δ'/δ can be written explicitly as a function of x:

$$\frac{\delta'}{\delta} = \sqrt{\frac{(r' - \epsilon' - x)^2 + (r^2 - x^2)}{(r' - x)^2 + (r^2 - x^2)}}. \tag{7}$$

We will compute a bound on the maximum of the above ratio for $x \in [-r, \rho]$. It is more convenient to consider

$$f(x) = \frac{(r' - \epsilon' - x)^2 + (r^2 - x^2)}{(r' - x)^2 + (r^2 - x^2)}. \tag{8}$$

We will prove that for $x \in [-r, \rho]$ we have

$$f(x) \leq \left(1 - \frac{\epsilon' r'}{r'^2 + r^2}\right). \tag{9}$$

We will consider two cases.

Case I. $r \leq r' - \epsilon'$. We first claim that in this case for any $x \in (0, \rho]$, we have

$$f(-x) \geq f(x). \tag{10}$$

It is easy to verify that the above is true if and only if

$$\epsilon' x \left(r'(r' - \epsilon') - r^2\right) \geq 0. \tag{11}$$

This holds true under the given assumption of the case. Thus it suffices to consider the maximum of $f(x)$ in the interval $[-r, 0]$. Differentiating $f(x)$ and simplifying its numerator, we get

$$2\epsilon' \left(\epsilon' r' - (r'^2 - r^2)\right). \tag{12}$$

This quantity never vanishes unless,

$$\epsilon' = \frac{(r'^2 - r^2)}{r'}. \tag{13}$$

It can be shown for this value of ϵ' both the numerator and the denominator of $f(x)$ have the same root,

$$x = \frac{r^2 + r'^2}{2r'} \geq r, \tag{14}$$

a value outside of the interval $[-r, 0]$. Thus, we only need to compare $f(0)$ and $f(-r)$. In fact since $\epsilon' \leq r' - r$, the quantity in (12) is negative, thus $f(x)$ is decreasing. Hence the maximum value of $f(x)$ on $[-r, \rho]$ is

$$f(-r) = \frac{(r' + r - \epsilon')^2}{(r' + r)^2} = \left(1 - \frac{\epsilon'}{r' + r}\right)^2. \tag{15}$$

We claim

$$f(-r) \leq \left(1 - \frac{\epsilon' r'}{r'^2 + r^2}\right). \tag{16}$$

Equivalently, we claim

$$\left(1 - \frac{\epsilon' r'}{r'^2 + r^2}\right) \geq \left(1 - \frac{\epsilon'}{r' + r}\right)^2 =$$

$$1 + \frac{\epsilon'^2}{(r' + r)^2} - \frac{2\epsilon'}{r' + r}. \tag{17}$$

Simplifying, this amounts to showing

$$\frac{2}{r' + r} \geq \frac{r'}{r'^2 + r^2} + \frac{\epsilon'}{(r' + r)^2}. \tag{18}$$

It is easy to show that

$$\frac{2r'}{(r'+r)^2} \geq \frac{r'}{r'^2+r^2}. \tag{19}$$

Finally, we can verify the following

$$\frac{2}{r'+r} \geq \frac{2r'}{(r'+r)^2} + \frac{\epsilon'}{(r'+r)^2}. \tag{20}$$

Thus we have proved (16).

Case II. $r > r' - \epsilon'$. By our previous statement on a root of $f'(x)$ and (14), and since in this case $\rho = r' - \epsilon'$, $f'(x)$ does not vanish on $[-r, \rho]$, so we only need to bound $f(r' - \epsilon')$. We claim

$$f(r' - \epsilon') = 1 - \frac{\epsilon'^2}{r^2 - r'^2 + 2r'\epsilon'} < (1 - \frac{\epsilon' r'}{r'^2 + r^2}). \tag{21}$$

This is equivalent to proving

$$\frac{r'}{r'^2 + r^2} < \frac{\epsilon'}{r^2 - r'^2 + 2r'\epsilon'}. \tag{22}$$

After simplification this is equivalent to

$$r'(r^2 - r'^2) < \epsilon'(r^2 - r'^2). \tag{23}$$

This is valid. Finally, substituting $\epsilon' = \epsilon^2/2r'$, we have proved (2), and using $r' \leq 2r$ in (2) we get (3). □

Corollary 3 *Let p, p', p'', v be as in Theorem 2 and set $R = \max\{d(p, v_i), i = 1, \ldots, n\}$. If $r' \leq 2r$, we have*

$$\delta' \leq \delta\sqrt{1 - \frac{\epsilon^2}{10R^2}} \leq \delta \exp\left(\frac{-\epsilon^2}{20R^2}\right). \tag{24}$$

Proof. The first inequality follows from Theorem 2 and the definition of R. To prove the next inequality, we use that $1 + x \leq \exp(x)$, and set $x = -\epsilon^2/10R^2$. □

4 The Blindfold Triangle Algorithm

In this section we describe a variation of the Triangle Algorithm described in [8] for solving the convex hull decision problem. As the Triangle Algorithm itself, given $p' \in \text{Conv}(S) - \{p\}$, the algorithm searches for a triangle $\triangle pp'v_j$ where $v_j \in S$ such that $d(p', v_j) \geq r_j$. Given such a triangle, the algorithm uses v_j as a pivot point to compute a new iterate $p'' \in \text{Conv}(S)$ such that $d(p'', p) < d(p', p)$. It replaces p'' with p' and repeats. Since the coordinates of p are not known, nor do we know if such prescribed distances correspond to an actual point, we will need to adjust the Triangle Algorithm to take conservative, but improving steps while making use of Theorem 2. For this reason we refer to this version as the *Blindfold Triangle Algorithm*.

The input to the algorithm is a prescribed tolerance $\epsilon > 0$, and a set of distances r_i, $i = 1, \ldots, n$, assumed to correspond to $d(p, v_i)$ for some point p. We assume to have selected an initial point $p' \in \text{Conv}(S)$. The Blindfold Triangle Algorithm is described as follows.

- **Step 1.** Pick any $p' \in \text{Conv}(S)$ (e.g. $p' = \frac{1}{n}\sum_{i=1}^{n} v_i$), check if

  $$|d(p', v_i) - r_i| \leq \epsilon, \quad \forall i = 1, \ldots, n.$$

  If the above holds, stop. We shall refer ro p' as an *ϵ-approximate solution*. Otherwise, go to Step 2.

- **Step 2.** Check if $d(p', v_i) > r_i$, $\forall i = 1, \ldots n$. If the above holds, stop. Otherwise, go to Step 3.

- **Step 3.** Check if there exists v_j such that $d(p', v_j) \geq r_j$. We shall refer to such v_j as the *pivot point*. If no such v_j exists, then $d(p', v_i) < d(p, v_i)$, for all $i = 1, \ldots, n$. Stop. Otherwise, go to Step 4.

- **Step 4.** If $r'_j = d(p', v_j) \geq 2r_j$, then set $p'' = v_j$. Replace p' with p'', go to Step 1. Otherwise, set p'' to be the point that takes a step of size $\epsilon' = \epsilon^2/2r'_j$ from p' in the direction of v_j. Replace p' with p'', go to Step 1. We refer to p'' as the *iterate*.

When p'' is not equal to v_j it is given by

$$p'' = \alpha p' + (1 - \alpha)v_j, \quad \alpha = 1 - \frac{\epsilon'}{r'_j}. \tag{25}$$

Since p'' is a convex combination of p' and v_j, it will remain in $\text{Conv}(S)$. First, we state a result that characterizes the stopping conditions in the algorithm.

Theorem 4 *The algorithm termination is categorized as follows:*

(1): If the algorithm terminates in Step 1, it has determined an ϵ-approximate solution.

(2): If the algorithm terminates in Step 2, no point p with prescribed distances exists.

(3): If the algorithm terminates in Step 3, no point p with prescribed distances belongs to $\text{Conv}(S)$.

Proof. Part (1) is clear from definition. The proof of (2) follows from the fact that if such point p existed, by Theorem 1 it could not belong to $\text{Conv}(S)$. Consider the Voronoi cell $V(p')$ with respect to the two point set $\{p, p'\}$. Analogous to the proof of Theorem 1 in the present article we can argue that $\overline{V}(p') = \{x : d(x, p') \leq d(x, p)\}$ must necessarily contain a v_j, hence $d(p', v_j) \leq d(p, v_j)$. But this contradicts the termination criterion of Step 2. Part (3), follows from Theorem 1. □

Next we state some basic properties of the algorithm to be used in its complexity analysis. The proof is omitted as it is straightforward and analogous to that in [8].

Proposition 5 *The algorithm satisfies:*

(i) In each iteration the step-size α lies in $(0, 1]$.

(ii) Given an explicit representation of p' as a convex combination of v_i's, p'' can also be explicitly written as a convex combination of v_i's.

(iii) Each iteration of the algorithm uses at most $O(mn)$ arithmetic operations and comparisons.

(iv) Each v_i can be selected as an iterate p'' at most once (see definition of iterate in Step 4).

(v) If the point p exists, and if v_j is selected as a pivot point more than once, then except possibly for its first selection as an iterate, in any subsequent selection of v_j as a pivot point the iterate p'' will satisfy $d(p, p'') \leq r_j$.

By part (iv) of Proposition 5, the number of iterations where an element $v_j \in S$ is selected as an iterate is at most n. Therefore, except for n iterations, in any other iteration of the algorithm the inequality $r' \leq 2r$ holds and thus inequality (3) in Theorem 2 holds so that (24) in Corollary 3 applies. The analysis of complexity of the Blindfold Triangle Algorithm will make repeated use of (24). For the sake of simplicity in the forgoing complexity analysis we will exclude the occurrence of exceptional iterations where $r' > 2r$ so that we assume (24) applies in every iteration.

Theorem 6 *Assume p is in $\mathrm{Conv}(S)$. Pick $p_0 \in \mathrm{Conv}(S) - \{p\}$, let $\delta_0 = d(p_0, p) \geq \epsilon$. Set $R = \max\{d(p, v_i), i = 1, \ldots, n\}$. Then the algorithm computes an ϵ-approximate solution in a finite number of iterations k_ϵ satisfying,*

$$k_\epsilon = \left\lceil \frac{20R^2}{\epsilon^2} \ln\left(\frac{\delta_0}{\epsilon}\right) \right\rceil = O\left(\epsilon^{-2} \ln(\frac{1}{\epsilon})\right).$$

Proof. From Corollary 3, given $p' \in \mathrm{Conv}(S)$ in an iteration, if the algorithm gets to compute p'', then if p with prescribed distances exists, we must have $d(p, p') > \epsilon$. Otherwise, from the triangle inequality we would have

$$|d(p', v_i) - d(p, v_i)| \leq \epsilon, \quad \forall i = 1, \ldots, n.$$

Thus, Corollary 3 applies and from its repeated application we have

$$\delta_k \leq \delta_{k-1} \exp\left(-\frac{\epsilon^2}{20R^2}\right) \leq \cdots \leq \delta_0 \exp\left(-k\frac{\epsilon^2}{20R^2}\right).$$

In order to satisfy $\delta_k \leq \epsilon$, it suffices to solve for the smallest $k = k_\epsilon$ satisfying

$$\delta_0 \exp\left(-k\frac{\epsilon^2}{20R^2}\right) \leq \epsilon.$$

This gives the claimed k_ϵ. □

Remark. In each iteration we can continue to use the same pivot point v_j so long as $d(p', v_j) \geq r_j$, making the search for a pivot as efficient as possible.

5 Estimation of the Gap

Suppose we have computed a point p' in the convex hull of a subset of $m + 1$ points in S, forming a full-dimensional simplex in $\mathbb{R}^m$. Without loss of generality assume these points are $v_0, \ldots, v_m$. Thus p' can be written as a convex combination of these points in a unique fashion. Also, the set of vectors $v_i - v_0$, $i = 1, \ldots, m$ forms a linearly independent set in $\mathbb{R}^m$. We wish to represent p and p' in terms of $v_0, \ldots, v_m$ and use it to estimate the gap $d(p, p')$, given the following

$$d(v_i, p) = r_i, \quad d(v_i, p') = r'_i, \quad i = 0, \ldots, m, \tag{26}$$

$$|r_i - r'_i| \leq \epsilon, \quad i = 0, \ldots, m. \tag{27}$$

Squaring the equations in (26), subtracting the first from the remaining m equations gives,

$$d^2(v_i, p) - d^2(v_0, p) = r_i^2 - r_0^2, \quad i = 1, \ldots, m, \tag{28}$$

$$d^2(v_i, p') - d^2(v_0, p') = r_i'^2 - r_0'^2, \quad i = 1, \ldots, m. \tag{29}$$

Equivalently, for $i = 1, \ldots, m$ this gives

$$(v_0 - v_i)^T p = \gamma_i, \quad (v_0 - v_i)^T p' = \gamma'_i, \tag{30}$$

where for $i = 1, \ldots, m$ we have

$$\gamma_i = \frac{1}{2}\left(r_i^2 - r_0^2 - d^2(v_i, 0) + d^2(v_0, 0)\right),$$

$$\gamma'_i = \frac{1}{2}\left(r_i'^2 - r_0'^2 - d^2(v_i, 0) + d^2(v_0, 0)\right). \tag{31}$$

Let W be the $m \times m$ matrix whose i-th row is

$$v_0^T - v_i^T, \quad i = 1, \ldots, m.$$

Let $\gamma = (\gamma_1, \ldots, \gamma_m)^T$, and $\gamma' = (\gamma'_1, \ldots, \gamma'_m)^T$. Then we have

$$Wp = \gamma, \quad Wp' = \gamma'.$$

This implies

$$p - p' = W^{-1}(\gamma - \gamma').$$

From (27), and assuming Δ is the diameter of $\mathrm{Conv}(S)$, we have

$$|r_i^2 - r_i'^2| \leq \epsilon(r_i + r'_i) \leq 2\epsilon\Delta, \quad i = 0, \ldots, m.$$

Thus for $i = 1, \ldots, m$ we have

$$|\gamma_i - \gamma'_i| \leq \frac{1}{2}\left(|r_i^2 - r_i'^2| + |r_0'^2 - r_0'^2|\right) \leq 2\epsilon\Delta.$$

Hence we conclude

$$d(\gamma, \gamma') \leq 2\sqrt{m}\Delta\epsilon,$$

implying the following bound on the gap

$$d(p, p') \leq \|W^{-1}\| 2\sqrt{m}\Delta\epsilon, \tag{32}$$

where $\|\cdot\|$ denotes the 2-norm of a matrix.

Remark. When the data is integral, there exists an ϵ_* such that if $d(p, p') \leq \epsilon_*$, p also lies in the convex hull of $v_0, \ldots, v_m$. This follows from the usual LP sensitivity. Thus, the algorithm can correctly solve the ambiguous convex hull problem when viewed as a decision problem.

6 Solving a Linear System as an Ambiguous Convex Hull Problem

Theorem 7 *Consider a linear system $Ax = b$, where A is an $m \times m$ invertible matrix. Let p be the solution to the equation. Assume we are given $v_0 \in \mathbb{R}^m$ and r_0 such that $d(p, v_0) = r_0$. Let $e = (1, \ldots, 1)^T \in \mathbb{R}^m$, let v_i be the i-th row of the matrix $V = ev_0^T - A$, and let $r_i = d(v_i, p)$, $i = 1, \ldots, m$. Then p is the unique solution to the set of equations*

$$d(v_i, p) = r_i, \quad i = 0, \ldots, m.$$

Moreover, for $i = 1, \ldots, m$ we have,

$$r_i^2 = 2b_i + r_0^2 + d^2(v_i, 0) - d^2(v_0, 0).$$

In particular, if $p \in \text{Conv}\{v_0, \ldots, v_m\}$, the Blindfold Triangle Algorithm can give an ϵ-approximate solution to $Ax = b$ in $O(m^2\epsilon^{-2}\ln(\epsilon^{-1}))$ arithmetic operations.

Proof. From the relationship between A, V, v_0, b, p, the definition of W and γ in (31) we have

$$Ap = b = Wp = (ev_0^T - V)p = \frac{1}{2}\gamma.$$

This completes the proof. □

Theorem 7 suggests interesting computational possibilities in using the Blindfold Triangle Algorithm to solve a linear system, given that the distance of the solution to a single point is known, e.g. given the Euclidean norm of the solution as is the case when A is an orthogonal matrix. Such an approach would compute an approximate solution faster than the traditional method of computing LU factorization.

7 Remarks

In this article we have presented a variation of the Triangle Algorithm for the convex hull decision problem, called the *Blindfold Triangle Algorithm*. It tries to determine if there exists a point p in the convex hull of a given set of points S, knowing only a set of distances, presumably corresponding to the points in S. In contrast with the Triangle Algorithm it takes smaller steps because it does not know the coordinates of p, nor does it know if such a point exists. Despite the conservative steps it is only slower than the Triangle Algorithm by a factor of $\ln(\epsilon^{-1})$. This could also mean that the Triangle Algorithm itself should have a better complexity (see [8]). An interesting application of the Blindfold Triangle Algorithm is in solving certain linear systems (e.g. orthogonal coefficient matrix), offering a new approach for solving this problem (see Theorem 7). We are optimistic that both the Triangle Algorithm as well as the Blindfold Triangle Algorithm will offer alternative algorithms for the convex hull decision problem, its variations, linear programming, and more. These algorithms suggest new lines of research in several different areas. We hope to carry out some computational testing as well as pursue theoretical and practical applications of the results.

Acknowledgements. I am grateful to three anonymous reviewers for a very careful reading of this article and for their helpful comments and suggestions that resulted in improvements. I also thank Iraj Kalantari for a discussion on the use of a bisecting hyperplane to prove Theorem 1, resulting in a simpler geometric proof than the one given in [8]. As suggested by a reviewer, it is also possible to give a direct proof Theorem 1, removing the embedded contradiction.

References

[1] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete Comput. Geom.*, 16(4):369–387, 1996.

[2] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10:377–409, 1993.

[3] V. Chvátal. *Linear Programming*. W.H. Freeman and Company, New York, 1983.

[4] K.L. Clarkson. More output-sensitive geometric algorithm. In Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, 695–702, 1994.

[5] J. E. Goodman, J. O'Rourke (Editors). *Handbook of Discrete and Computational Geometry*, 2nd Edition (Discrete Mathematics and Its Applications), Chapman & Hall Boca Raton, 2004.

[6] Y. Jin and B. Kalantari. A procedure of Chvátal for testing feasibility in linear programming and matrix scaling. *Linear Algebra and its Applications*, 416:795–798, 2006.

[7] B. Kalantari. Canonical problems for quadratic programming and projective methods for their solution. *Contemporary Mathematics*, 114:243–263, 1990.

[8] B. Kalantari. A characterization theorem and an algorithm for a convex hull problem. arXiv:1204.1873v1, 2012.

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

[10] 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.

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

[12] L. Khachiyan and B. Kalantari. Diagonal matrix scaling and linear programming. *SIAM J. Optim.*, 4:668–672, 1992.