Annals of Operations Research (2020) 295:37–73
https://doi.org/10.1007/s10479-020-03557-0

ORIGINAL RESEARCH

Robust vertex enumeration for convex hulls in high dimensions

Pranjal Awasthi[1] · Bahman Kalantari[1] · Yikai Zhang[1]

Published online: 13 March 2020

Abstract
The problem of computing the vertices of the convex hull of a given input set $S = \{v_i \in \mathbb{R}^m : i = 1, \ldots, n\}$ is a classic and fundamental problem, studied in the context of computational geometry, linear and convex programming, machine learning and more. In this article we present *All Vertex Triangle Algorithm* (AVTA), a robust and efficient algorithm for this problem. On the one hand, without any assumptions, AVTA computes approximation to the subset $\overline{S}$ of all K vertices of the convex hull of S so that the convex hull of the approximate subset of vertices is as close to $conv(S)$ as desired. On the other hand, assuming a known lower bound γ on the ratio Γ_*/R, where Γ_* the minimum of the distances from each vertex to the convex hull of the remaining vertices and R the diameter of S, AVTA can recover all of $\overline{S}$. Furthermore, assuming that instead of S the input is an ε-perturbation of S, $\overline{S}_\varepsilon$, where $\|v_i - v_i^\varepsilon\| \leq \varepsilon R$, AVTA can compute approximation to $conv(\overline{S}_\varepsilon)$, to any prescribed accuracy. Also, given a lower bound to the ratio Σ_*/R, where Σ_* is the minimum of the distances from each vertex to the convex hull of the remaining point of S, AVTA can recover all of $\overline{S}_\varepsilon$. We show $\Sigma_* \geq \rho_* \Gamma_*/R$, where ρ_* is the minimum distance between distinct pair of points in S and prove the following main results:

(1) Given any $t \in (0, 1)$, AVTA computes a subset $\overline{S}^t$ of $\overline{S}$ of cardinality $K^{(t)}$ in $O(nK^{(t)}(m + t^{-2}))$ operations so that for any $p \in conv(S)$ its Euclidean distance to $conv(\overline{S}^t)$ is at most tR.
(2) Given $\gamma \leq \gamma_* = \Gamma_*/R$, AVTA computes $\overline{S}$ in $O(nK(m + \gamma^{-2}))$ operations.
(3) If K is known, the complexity of AVTA is $O(nK(m + \gamma_*^{-2}) \log(\gamma_*^{-1}))$.

Assuming instead of S, its ε-perturbation, S_ε is given, we prove

(i) Given any $t \in (0, 1)$, AVTA computes a subset $\overline{S}_\varepsilon^t \subset \overline{S}_\varepsilon$ of cardinality $K_\varepsilon^{(t)}$ in $O(nK_\varepsilon^{(t)}(m + t^{-2}))$ operations so that for any $p \in conv(S)$ its distance to $conv(\overline{S}_\varepsilon^t)$ is at most $(t + \varepsilon)R$.
(ii) Given $\sigma \in [4\varepsilon, \sigma_* = \Gamma_*/R]$, AVTA computes $\overline{S}_\varepsilon$ in $O(nK_\varepsilon(m + \sigma^{-2}))$ operations, where $K \leq K_\varepsilon \leq n$.

A conference version of the article appeared in the Proceedings of AISTATS 2018.

✉ Bahman Kalantari
kalantar@cs.rutgers.edu

Extended author information available on the last page of the article

(iii) If $\gamma \leq \gamma_* = \Gamma_*/R$ is known satisfying $4\varepsilon \leq \gamma\rho_*/R$, $AVTA$ computes $\overline{S}_\varepsilon$ in $O(nK_\varepsilon(m + (\gamma\rho_*)^{-2}))$ operations.
(iv) Given $\sigma \in [4\varepsilon, \sigma_*]$, if K is known, AVTA computes $\overline{S}_\varepsilon$ in $O(nK(m+\sigma_*^{-2})\log(\sigma_*^{-1}))$ operations.

We also consider the application of AVTA in the recovery of vertices through the projection of S or S_ε under a Johnson–Lindenstrauss randomized linear projection $L : \mathbb{R}^m \to \mathbb{R}^{m'}$. Denoting $U = L(S)$ and $U_\varepsilon = L(S_\varepsilon)$, by relating the robustness parameters of $conv(U)$ and $conv(U_\varepsilon)$ to those of $conv(S)$ and $conv(S_\varepsilon)$, we derive analogous complexity bounds for probabilistic computation of the vertex set of $conv(U)$ or those of $conv(U_\varepsilon)$, or an approximation to them. Finally, we apply AVTA to design new practical algorithms for two popular machine learning problems: topic modeling and non-negative matrix factorization. For topic models, our new algorithm leads to significantly better reconstruction of the topic-word matrix than state of the art approaches of Arora et al. (International conference on machine learning, pp 280–288, 2013) and Bansal et al. (Advances in neural information processing systems, pp 1997–2005, 2014). Additionally, we provide a robust analysis of AVTA and empirically demonstrate that it can handle larger amounts of noise than existing methods. For non-negative matrix factorization we show that AVTA is competitive with existing methods that are specialized for this task in Arora et al. (Proceedings of the forty-fourth annual ACM symposium on theory of computing, ACM, pp 145–162, 2012a). We also contrast AVTA with Blum et al. (Proceedings of the twenty-seventh annual ACM-SIAM symposium on discrete algorithms, Society for Industrial and Applied Mathematics, pp 548–557, 2016) *Greedy Clustering* coreset algorithm for computing approximation to the set of vertices and argue that not only there are regimes where AVTA outperforms that algorithm but it can also be used as a pre-processing step to their algorithm. Thus the two algorithms in fact complement each other.

Keywords Convex hull membership · Approximation algorithms · Machine learning · Linear programming · Random projections

1 Introduction

In this article we present a robust and efficient algorithm called *All Vertex Triangle Algorithm* (AVTA). Given input set $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, on the one hand without any assumptions, AVTA computes approximation to the subset $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ of all vertices of $conv(S)$ so that the convex hull of the approximate subset of vertices is as close to $conv(S)$ as desired. Specifically, given any $t \in (0, 1)$, AVTA computes a subset $\overline{S}^t$ of $\overline{S}$ so that the distance from any point $p \in conv(S)$ to $conv(\overline{S}^t)$ is at tR, where R is the diameter of S. On the other hand, under the assumption that we are given a constant $\gamma \leq \Gamma_*/R$, where Γ_* is the minimum of the distances from each vertex to the convex hull of the remaining vertices, it can compute all of $\overline{S}$. Furthermore, assuming that instead of S the input is an ε-perturbation of S, $\overline{S}_\varepsilon$, where $\|v_i - v_i^\varepsilon\| \leq \varepsilon R$, AVTA can compute approximation to the convex hull of $\overline{S}_\varepsilon$, to any prescribed accuracy.

AVTA, a *fully polynomial-time approximation scheme*, builds upon the *Triangle Algorithm* (Kalantari 2015), designed to solve the *the convex hull membership problem*. Specifically, given S, the Triangle Algorithm tests if a distinguished point p lies in the $conv(S)$, either by computing a point $p_\varepsilon \in conv(S)$ to within a prescribed distance to p, or a hyperplane that separates p from $conv(S)$. Before describing AVTA and its applications we wish to give

an overview of the related problems and research, as well as their history, significance and connections to our work.

The convex hull membership problem is a basic problem in computational geometry and a very special case of the *convex hull problem*, see Toth et al. (2004). Besides being a fundamental problem in computational geometry, it is a basic problem in linear programming (LP). In fact LP with integer coefficients can be reduced to a convex hull membership problem. Furthermore, the two most famous polynomial-time LP algorithms, the ellipsoid algorithm of Khachiyan (1980) and the projective algorithm of Kalantari (1984), are in fact explicitly or implicitly designed to solve the convex hull membership problem when $p = 0$, see Jin and Kalantari (2006). Furthermore, using an approach suggested by Jin and Kalantari (2006) shows a direct connection between a general LP feasibility and this homogeneous case of the convex hull membership problem.

An important problem in computational geometry and machine learning is the *irredundancy problem*, the problem of computing all the vertices of $conv(S)$, see Toth et al. (2004). Clearly, any algorithm for LP feasibility can be used to solve the irredundancy problem by solving a sequence of $O(n)$ convex hull membership problems. For results that reduce the number of linear programming problems, see e.g. Clarkson (1994) and Chan (1996a, b). Some applications require the description of $conv(S)$ in terms of its vertices, facets and adjacencies, see Chazelle (1993). The complexity of many exact algorithms for irredundancy of finite points set is exponential in terms of the dimension of the points, thus only practical in very low dimensions. On the other hand, the convex hull membership problem by itself has been studied in the context of large scale applications where simplex method or polynomial time algorithms are too expensive to run. Thus approximation schemes have been studied for the problem.

Not only is convex hull detection a fundamental problem in computational geometry, state of the art algorithms for many machine learning problems rely on being able to solve this problem efficiently. Consider for instance the problem of non-negative matrix factorization (NMF) (Lee and Seung 2001). Here, given access to a data matrix A, we want to compute non-negative, low rank matrices U and V such that $A = UV$. Although in general this problem is intractable, recent results show that under a natural *separability* assumption, see Donoho and Stodden (2003), such a factorization can be computed efficiently, see Arora et al. (2012a). The key insight in these works is that under the separability assumption, the rows of the matrix V will appear among the rows of A. Furthermore, the rows of V will be the *vertices* of the convex hull of rows of A. Hence, a fast algorithm for detecting the vertices will lead to a fast factorization algorithm as well.

The organization of the the article is as follows. In Sect. 2 we mention and review related work on the irredundancy problem. In Sect. 3 we describe the high level idea of the algorithm and introduce the main results. In Sect. 4, we describe the convex hull membership oracle Triangle Algorithm. This will be used throughout the the article. The efficient implementation of Triangle Algorithm is described in the "Appendix". In Sect. 5, we describe *All Vertex Triangle Algorithm* (AVTA), a Algorithm, for computing all vertices of the convex hull of a given finite set of points, S. We discuss several applications of this, in particular in solving the convex hull membership problem itself. Other applications will be described in subsequent sections. In Sect. 6, we consider the performance of AVTA under perturbation of data. In Sect. 7, we consider AVTA with Johnson–Lindenstrauss projections. Furthermore, we consider the performance of AVTA under perturbation of data with Johnson–Lindenstrauss projections.

2 Related work

Convex hull membership query oracles The convex hull membership problem can be formulated as the minimization of a convex quadratic function over the unit simplex. This particular convex program finds applications in statistics, approximation theory, and machine learning, see e.g Clarkson (2010) and Zhang (2003) who consider the analysis of a greedy algorithm for minimizing smooth convex functions over the unit simplex. The algorithm of Frank and Wolfe (1956) is a classic greedy algorithm for convex programming. When the the convex hull of a set of points does not contain the origin, the problem of computing the point in the convex hull with least norm, known as *polytope distance* is also a problem of interest. In some applications the polytope distance refers to the distance between two convex hulls, a fundamental problem in machine learning, known as SVM, see e.g. Burges (1998). The algorithm of Gilbert (1966) for the polytope distance problem is one of the earliest known algorithms. Gärtner and Jaggi (2009) show Gilbert's algorithm coincides with Frank–Wolfe algorithm when applied to the minimization of a convex quadratic function over a unit simplex. In this case the algorithm is known as *sparse greedy approximation*. For many results regarding the applications of the minimization of a quadratic function over a simplex, see Zhang (2003), Clarkson (2010) and Gärtner and Jaggi (2009). Clarkson (2010) analyzes the Frank–Wolfe and its variations while studying the notion of *coresets*. While the Triangle Algorithm has features that are very similar to those of Frank–Wolfe algorithm, there are other features and properties that make it an algorithm distinct from Frank–Wolfe or Gilbert's algorithm. To describe these differences, consider the distance between p and $conv(S)$: $\Delta = \min\{d(p', p) \equiv \|p' - p\| : \quad p' \in conv(S)\} = d(p_*, p)$.

Clearly, $p \notin conv(S)$, if and only if $\Delta > 0$. The goal of the convex hull membership problems (equivalently an LP feasibility) is to test feasibility, i.e. if p lies in $conv(S)$. Solving this does not require the computation of Δ when it is positive. Thus the goal of solving the convex hull membership is different from that of computing this distance $\Delta$when positive. When $p \in conv(S)$, the analysis of complexity of the Triangle Algorithm is essentially identical with Clarkson (2010) analysis of the basic Frank–Wolfe algorithm. Gärtner and Jaggi (2009) on the other hand analyze the complexity of Gilbert's algorithm for the polytope distance problem, i.e. the approximation of Δ, however under the assumption that $\Delta > 0$. Gärtner and Jaggi (2009) do not address the case when $\Delta = 0$.

What distinguishes the Triangle Algorithm from the Frank–Wolfe Algorithm and Gilbert's algorithms is the *distance dualities* which gives more flexibility to the algorithm. The algorithm we will analyze in this article, namely AVTA, is designed to generate vertices of $conv(S)$. It makes repeated use of the distance dualities of the Triangle Algorithm, resulting in an over all efficient algorithm for computing the vertices of $conv(S)$, or very good approximation to these vertices, even under perturbation of the input set. Indeed AVTA is testimonial to the uniqueness of the Triangle Algorithm while itself is a nontrivial extension of the Triangle Algorithm. AVTA finds many applications in computational geometry and machine learning. Some of these are demonstrated here theoretically and computationally. We next describe AVTA in more detail.

Irredundancy problem in machine learning A problem related to the irredundancy problem is known as *topic modeling* (Blei 2012). Here one is given access to a large corpus of documents, with each document represented as a long vector consisting of frequency in the document of every word in the vocabulary. This is known as the bag-of-words representation. Each document is assumed to represent a mixture of up to K hidden topics. A popular generative model for such documents is the following: For every document d, a K dimensional

vector θ_d is drawn from a distribution over the simplex. Typically this distribution is the Dirichlet distribution. Then, for each word in the document, a topic is chosen according to θ_d. Finally, given a chosen topic i, a word is output according to the topic distribution vector β_i. This is known as the Latent Dirichlet Allocation (LDA) model (Blei et al. 2003). The parameters of this model consist of the topic-word matrix β so that β_i defines the distribution over words for topic i. Additionally, there are hyper parameters associated with the Dirichlet distributions generating the topic distribution vector θ_d. The topic modeling problem concerns learning the topic-word matrix β and the parameters of the topic generating distribution. Similar to NMF, the problem is intractable in the worst case but can be efficiently solved under *separability* (Arora et al. 2012b). In this context, the separability assumption requires that for each topic i, there exists an *anchor word* that has a non-zero probability of occurring only under topic i. Separability is an assumption that is known to hold for real world documents (Arora et al. 2012b). The key component towards learning the model parameters is a fast algorithm for finding the anchor words. The algorithm of Arora et al. (2012b, 2013) uses the word-word covariance matrix and shows that under separability, the vertices of the convex hull of the rows of the matrix will correspond to the anchor words. Similarly, the work of Ding et al. (2013) shows that finding the vertices of the convex hull of the document-word matrix will also lead to detection of anchor words. Both approaches rely on the vertex detection subroutine. Furthermore, in the case of topic models, the documents are limited in size and this translates to the fact that one is given a perturbation of the set S. The goal is to use this perturbed set to approximate the original vertices $\overline{S}$. Hence in this application it is crucial that the approach to finding the vertices be robust to noise. Following the setting of Arora et al. (2013) we analyze AVTA under the setting of perturbed irredundancy problem. We show AVTA is able to find the 'robust ' vertices with guarantees that is related to the geometric properties of given set.

Approximation version of irredundancy problem In addition to the perturbed irredundancy problem, Blum et al. (2016) consider the approximation version of the irredundancy problem where a bi-criterion algorithm is proposed based on *Nearest Neighbor Oracle*, called *Greedy Clustering*, for computing a subset of vertices T satisfying two properties: (i) the Hausdorff distance between $conv(T)$ and $conv(S)$ is bounded above by $(8t + t^2)R$; (ii) $|T| = O(K_{opt}/t^2)$ where K_{opt} is the cardinality of smallest set of points which attains t approximation quality. Since $T \subset S$, this implies that $t = \Omega((K_{opt}/n)^{1/2})$. The running time of the algorithm is $O\left(\frac{nK_{opt}}{t^2}\left(m + \frac{K_{opt}}{t^8} + \frac{K_{opt}^2}{t^4}\right)\right)$. While there is a theoretical bound on the size of T as a polynomial in $1/t$, it is inefficient since it uses the *Nearest Neighbor Oracle*. Indeed, in AVTA, the Triangle Algorithm works as an approximate oracle which achieves great improvement in efficiency. Given that $\gamma_* = \Gamma_*/R$ is $\Omega(t)$, we cannot use fewer than $|\overline{S}|$ vertices to give an t approximation. This argument shows that in Blum et al. (2016) $K_{opt} = |\overline{S}|$. In a general case where γ_* is arbitrarily close to 0, AVTA will find all vertices in $O(nK_t(m + \frac{1}{t^2}))$ time where $K_t = |\overline{S}^t|$. While we so far have no nontrivial bound on K_t, it is known that $K_t \leq n$. In this case the complexity of AVTA is $O(n^2m + n^2/t^2)$ and Greedy Clustering requires at least $O(nm/t^2 + n/t^{10})$ to achieve the same accuracy. It could be concluded that there exists regimes that AVTA outperforms Greedy Clustering. It is interesting to observe that AVTA could be used as a pre-processing algorithm for Greedy Clustering. By our analysis, AVTA only detects vertices and will not omit any of them. In case $n >> K_t$, we can use AVTA to delete points inside the convex hull thus reduce the size of the problem for Greedy Clustering. In summary, the two algorithms complement each other.

3 Main results

In this section we describe the high level idea and introduce the main result about AVTA. To describes, AVTA we need to define some parameters. We say $conv(S)$ is Γ_*-*robust*, if Γ_* is the minimum of the distances from each $\overline{v}_i \in \overline{S}$ to $conv(\overline{S}\backslash\{\overline{v}_i\})$. Set $R = \max\{d(v_i, v_j), v_i, v_i \in S\}$, the diameter of S. The AVTA works as follows:

AVTA (informal)

Step 0. Find one vertex in S(which is easy to do) and add such vertex to working set $\widehat{S}$.
Step 1. Randomly select a point $v \in S\backslash\widehat{S}$.
Step 2. Call membership oracle to test if $v \in conv(\widehat{S})$.
Step 3. If $v \notin conv(\widehat{S})$, use the separation hyperplane that the oracle found to capture a new vertex.
Step 4. If $v \in conv(\widehat{S})$, remove v from S.
Step 5. Repeat above step until S is empty.

(1) Given any $t \in (0, 1)$, AVTA can compute a subset $\overline{S}^t$ of $\overline{S}$ so that the distance of each point in $conv(S)$ to $conv(\overline{S}^t)$ is at most tR. The corresponding number of operations is

$$O(nK^{(t)}(m + t^{-2})), \quad K^{(t)} = |\overline{S}^t|. \tag{1}$$

(2) If a number $0 < \gamma \leq \Gamma_*/R$ is known, the number of operations of $AVTA$ to computes $\overline{S}$ is.

$$O(nK(m + \gamma^{-2})). \tag{2}$$

(3) If only K is known, the number of operations of $AVTA$ to compute $\overline{S}$ is

$$O(nK(m + \gamma_*^{-2})) \log \gamma_*^{-1}). \tag{3}$$

In practice the input set may be not S but a perturbation of it, $S_\varepsilon = \{v_1^\varepsilon, \ldots, v_n^\varepsilon\}$, where $\|v_i - v_i^\varepsilon\| \leq \varepsilon R$. The set of perturbed vertices, $\overline{S}_\varepsilon = \{\overline{v}_1^\varepsilon, \ldots, \overline{v}_K^\varepsilon\}$ may differ considerably from the set of actual vertices of $conv(S_\varepsilon)$. Under a mild assumption on ε, AVTA computes $\overline{S}_\varepsilon = \{\overline{v}_1^\varepsilon, \ldots, \overline{v}_K^\varepsilon\}$. More generally, given any $t \in (0, 1)$, AVTA computes a subset $\overline{S}_\varepsilon^t$ of $\overline{S}_\varepsilon$ so that the distance from any $p \in conv(S)$ to $conv(\overline{S}_\varepsilon^t)$ is at most $(t + \varepsilon)R$. The complexity of AVTA for this variation of the problem is analogous to the unperturbed case, however it makes use a weaker parameter. We say $conv(S)$ is Σ_*-weakly robust, if Σ_* is the minimum of the distances of each vertex in S to the convex hull of all the remaining points in S. In Fig. 1 we show a simple example where Γ_* and Σ_* are shown for set of eight points.

We first prove when $\sigma_* = \Sigma_*/R \geq 4\varepsilon$, $\overline{S}_\varepsilon$ is a subset of vertices of $conv(S_\varepsilon)$ and $conv(S_\varepsilon)$ is at least $\Sigma_*/2$-weakly robust. Using this, we prove

(i) Given any $t \in (0, 1)$, AVTA can compute a subset $\overline{S}_\varepsilon^t$ of $\overline{S}_\varepsilon$ so that the distance from each p in $conv(S)$ to $conv(\overline{S}_\varepsilon^t)$ is at most $(t + \varepsilon)R$. The corresponding number of operations is

$$O(nK_\varepsilon^t(m + t^{-2})), \quad K_\varepsilon^t = |\overline{S}_\varepsilon^t|. \tag{4}$$

(ii) If $\sigma \leq \sigma_* = \Sigma_*/R$ is known to satisfy $4\varepsilon \leq \sigma$, the number of operations of $AVTA$ to computes $\overline{S}_\varepsilon$, is.

$$O(nK_\varepsilon(m + \sigma^{-2})), \tag{5}$$

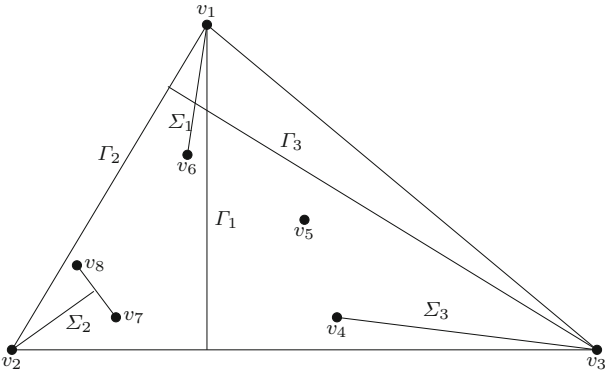

Fig. 1 $\Gamma_* = \Gamma_1$ and $\Sigma_* = \Sigma_2$

where K_ε is at most the cardinality of the set of vertices of S_ε.

Clearly $\Gamma_* \geq \Sigma_*$, however we prove

$$\Sigma_* \geq \frac{\Gamma_*}{R}\rho_*, \tag{6}$$

where ρ_* is the minimum distance between distinct pair of points in S. This allows deriving a lower bound to Σ_* from a known lower bound on Γ_*. Thus we can alternatively write

(iii) If $\gamma \leq \gamma_* = \Gamma_*/R$ is known satisfying $4\varepsilon \leq \gamma\rho_*/R$, the number of operations of $AVTA$ to compute $\overline{S}_\varepsilon$ is.

$$O(nK_\varepsilon(m + (\gamma\rho_*)^{-2})). \tag{7}$$

(iv) If only K is known, where $4\varepsilon \leq \sigma_* = \Sigma_*/R$, the number of operations of $AVTA$ to computes $\overline{S}_\varepsilon$ is.

$$O(nK_\varepsilon(m + \sigma_*^{-2})\log(\sigma_*^{-1})). \tag{8}$$

We also consider the application of AVTA in the recovery of vertices through the projection of S or S_ε under a Johnson–Lindenstrauss randomized linear projection $L : \mathbb{R}^m \to \mathbb{R}^{m'}$. By relating the robustness parameters of $conv(U)$ and $conv(U_\varepsilon)$, where $U = L(S)$ and $U_\varepsilon = L(S_\varepsilon)$, to those of $conv(S)$ and $conv(S_\varepsilon)$, we derive analogous complexity bounds for probabilistic computation of the vertex set of $conv(U)$ or those of $conv(U_\varepsilon)$, or an approximation to these subsets for a given $t \in (0, 1)$. Table 1 summarizes the complexities of computing desired sets under various cases.

4 Triangle Algorithm: efficient membership oracle

The *Triangle Algorithm* described in Kalantari (2015) is a simple iterative algorithm for solving the *convex hull membership problem*, a fundamental problem in linear programming and computational geometry. Formally, the convex hull membership problem is as follows: Given a set of point $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$ and a distinguished point $p \in \mathbb{R}^m$, test if $p \in conv(S)$. If $p \notin conv(S)$, find a hyperplane that separates p from $conv(S)$. If $p \in conv(S)$, the Triangle Algorithm solves the problem to within prescribed precision by generating a sequence of points inside of $conv(S)$ that get sufficiently close to p.

Table 1 $\Gamma_* = \min\{d(\overline{v}_i, conv(\overline{S}\backslash\{\overline{v}_i\}))\}$, $\Sigma_* = \min\{d(\overline{v}_i, conv(S\backslash\{\overline{v}_i\}))\}$, $\rho_* = \min\{d(v_i, v_j), i \neq j\}$

Input and description	Computed via AVTA	Conditions	Complexity
$S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$	$\overline{S}$, vertices of $conv(S)$, $\lvert\overline{S}\rvert = K$	$\gamma \leq \gamma_* \equiv \Gamma_*/R$ is known	$O\big(nK(m+\gamma^{-2})\big)$
$R = \max\{\lVert v_i - v_j\rVert : v_i, v_j \in S\}$	$\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$	Only K is known	$O\big(nK(m+\gamma_*^{-2})\big) \times \log(\gamma_*^{-1})$
	Given $t \in (0,1)$, $\overline{S}^t \subset \overline{S}$, $\lvert\overline{S}^t\rvert = K^{(t)}$	General case	$O\big(nK^{(t)}(m+t^{-2})\big)$
	$d(p, conv(\overline{S}^t) \leq tR, \forall p \in conv(S)$		
$S_\varepsilon = \{v_1^\varepsilon, \ldots, v_n^\varepsilon\}$, a perturbation of S	$\widehat{S}_\varepsilon$, vertices in $conv(S_\varepsilon)$, $\lvert\widehat{S}_\varepsilon\rvert = K_\varepsilon$, $\overline{S}_\varepsilon = \{\overline{v}_1^\varepsilon, \ldots, \overline{v}_K^\varepsilon\} \subset \widehat{S}_\varepsilon$	$\sigma \leq \sigma_* \equiv \Sigma_*/R$ is known, $\varepsilon \leq \sigma/4$ $\gamma \leq \gamma_*$ is known, $\varepsilon \leq \gamma\rho_*/4R$	$O\big(nK_\varepsilon(m+\sigma^{-2})\big)$ $O\big(nK_\varepsilon(m+R^2/(\gamma\rho_*)^2)\big)$
		Only K is known, $\varepsilon \leq \sigma_*/4$	$O\big(nK_\varepsilon(m+\sigma_*^{-2})\big) \times \log(\sigma_*^{-1})$
$\lVert v_i^\varepsilon - v_i\rVert \leq \varepsilon R$	Given $t \in (0,1)$, $\overline{S}_\varepsilon^t \subset \overline{S}_\varepsilon$, $\lvert\overline{S}_\varepsilon^t\rvert = K_\varepsilon^{(t)}$	General case	$O\big(nK_\varepsilon^{(t)}(m+t^{-2})\big)$
	$d(p, conv(\overline{S}_\varepsilon^t) \leq (t+\varepsilon)R, \forall p \in conv(S)$		
J–L Projection of S	$\overline{U}$, vertices of $conv(U)$, $\lvert\overline{U}\rvert = K_{\varepsilon'}$	$\gamma \leq \gamma_*$ is known	$O\big(nK_{\varepsilon'}(m'+(\gamma(1-\varepsilon'))^{-2})\big)$
$U = L(S) = \{u_1, \ldots, u_n\}$	$\overline{U} = \{\overline{u}_1, \ldots, \overline{u}_{K_{\varepsilon'}}\}$, $\overline{U} \subset L(\overline{S})$	$m' = \varepsilon'^2/c\log n < m$, c a constant	
$u_i = L(v_i)$, $L : \mathbb{R}^m \to \mathbb{R}^{m'}$	Given $t \in (0,1)$, $\overline{U}^t \subset L(\overline{S})$, $\lvert\overline{U}^t\rvert = K_{\varepsilon'}^t$	General case	$O\big(nK_{\varepsilon'}^t(m'+t^{-2})\big)$
R' diameter of U	$d(q, conv(\overline{U}^t) \leq tR', \forall q \in conv(U)$		
J-L Projection of S_ε	$\widehat{U}_\varepsilon$, vertices in $conv(U_\varepsilon)$, $\lvert\widehat{U}_\varepsilon\rvert = K_{\varepsilon\varepsilon'}$,	$\sigma \leq \sigma_*$ is known, $\varepsilon \leq \sigma(1-\varepsilon')/4$	$O\big(nK_{\varepsilon\varepsilon'}(m'+(\sigma(1-\varepsilon'))^{-2})\big)$
$U_\varepsilon = L(S_\varepsilon) = \{u_1^\varepsilon, \ldots, u_n^\varepsilon\}$	$\overline{U}_\varepsilon = \{\overline{u}_1^\varepsilon, \ldots, \overline{u}_{K_*}^\varepsilon\}$, $\overline{U}_\varepsilon \subset \widehat{U}_\varepsilon$	$\gamma \leq \gamma_*$ is known, $\varepsilon \leq \gamma\rho_*(1-\varepsilon')/4R$	$O\big(nK_{\varepsilon\varepsilon'}(m'+R^2/(\gamma\rho_*(1-\varepsilon'))^2)\big)$
$u_i^\varepsilon = L(v_i^\varepsilon)$	Given $t \in (0,1)$, $\overline{U}_\varepsilon^t \subset \overline{U}_\varepsilon$, $\lvert\overline{U}_\varepsilon^t\rvert = K_{\varepsilon\varepsilon'}^{(t)}$	General Case	$O\big(nK_{\varepsilon\varepsilon}^{(t)}(m'+t^{-2})\big)$
	$d(q, conv(\overline{U}_\varepsilon^t) \leq (t+\varepsilon)R, \forall q \in conv(U)$		

Definition 1 Given p, S and $\varepsilon \in (0, 1)$, $p' \in conv(S)$ is an *ε-approximate solution* if for some $v \in S$,

$$d(p', p) \leq \varepsilon R, \quad R = \max\{d(v_i, v_j) : v_i, v_j \in S\}. \tag{9}$$

Given S, p and ε, **Triangle Algorithm**(S, p, ε) either computes an ε-approximate solution, or it computes a hyperplane separating p from $conv(S)$. AVTA to be described later will make repeated use of the this procedure. AVTA may use any other membership oracle that is capable of these tasks. Since the complexities for AVTA will be based on that of the Triangle Algorithm, we briefly describe it and its basic complexities. In the "Appendix" we describe a more efficient complexity. Given $u, v \in \mathbb{R}^m$, we interchangeably use $d(u, v) = \|u - v\|$. Given a point in $conv(S)$, the Triangle Algorithm searches for a *pivot* to get closer to p:

Definition 2 Given $p' \in conv(S)$, called *iterate*, we call $v \in S$ a *p-pivot* (or simply *pivot*) if

$$d(p', v) \geq d(p, v) \iff v^T p - v^T p' \geq \frac{1}{2}(\|p\|^2 - \|p'\|^2). \tag{10}$$

Definition 3 Given p, $p' \in conv(S)$ is a *p-witness* (or simply *witness*) if the orthogonal bisecting hyperplane to the line segment pp' separates p from $conv(S)$. Equivalently, $p' \in conv(S)$ is a *p-witness* if and only if $d(p', v_i) < d(p, v_i)$, for all $i = 1, \ldots, n$.

Given a point $p' \in conv(S)$ that is neither an ε-approximate solution nor a witness, the Triangle Algorithm finds a p-pivot $v \in S$. Then on the line segment $p'v$ it compute the closest point to p, denoted by $Nearest(p; p'v)$. It then replaces p' with $Nearest(p; p'v)$ and repeats the process. It is easy to show the following,

Proposition 1 *If an iterate $p' \in conv(S)$ satisfies $d(p', p) \leq \min\{d(p, v_i) : i = 1, \ldots, n\}$, and v_j is a p-pivot, then the new iterate*

$$p'' = Nearest(p; p'v_j) = (1 - \alpha)p' + \alpha v_j, \tag{11}$$

$$\alpha = \frac{(p - p')^T(v_j - p')}{d^2(v_j, p')} = \frac{p^T v_j - p'^T v_j - p^T p' + \|p'\|^2}{\|v_j\|^2 - 2p'^T v_j + \|p'\|^2}. \tag{12}$$

If

$$p' = \sum_{i=1}^{n} \alpha_i v_i, \quad \sum_{i=1}^{n} \alpha_i = 1, \quad \alpha_i \geq 0, \quad \forall i, \tag{13}$$

then

$$p'' = \sum_{i=1}^{n} \alpha'_i v_i, \quad \alpha'_j = (1 - \alpha)\alpha_j + \alpha, \quad \alpha'_i = (1 - \alpha)\alpha_i, \quad \forall i \neq j. \tag{14}$$

The correctness of the iterative step of the Triangle Algorithm is due to the following:

Theorem 1 (Distance duality) *Exclusively, either for each $p' \in conv(S)$ there exists $v_j \in S$ that is p-pivot, or there exists $p' \in conv(S)$ that is p-witness, i.e. $d(p', v_i) < d(p, v_i)$.* □

Theorem 2 *Given $\varepsilon > 0$, in $O(1/\varepsilon^2)$ iterations the Triangle Algorithm computes a ε-approximate solution, or it computes a witness. In particular, if $p \notin conv(S)$ and $\Delta = \min\{d(x, p) : x \in conv(S)\}$, the number of iterations to compute a witness is $O(R^2/\Delta^2)$.* □

The straightforward implementation of each iteration of the Triangle Algorithm is easily seen to take $O(mn)$ arithmetic operations. The algorithm can be described as follows:

Triangle Algorithm (S, p, $\varepsilon \in (0,1)$)

Step 0. Set $p' = \arg\min\{d(v_i, p) : v_i \in S\}$.
Step 1. If $d(p', p) \leq \varepsilon R$, then output p' as an ε-approximate solution, stop.
Step 2. If a p-pivot $v \in S$ exists, set $p' \leftarrow Nearest(p; p'v)$. Goto Step 1.
Step 3. Output p' as p-witness. Stop.

Remark 1 In each iteration of the Triangle Algorithm it suffices to have a representation of the iterate p' in terms of v_i's, i.e. $p' = \sum_{i=1}^{n} \alpha_i v_i$, where $\sum_{i=1}^{n} \alpha_i = 1$, $\alpha_i \geq 0$ for all $i = 1, \ldots, n$. It is not necessary to know the coordinates of p'. Rather it is enough to have an array of size n to store the vector of α_i's. Then assuming that we have stored $p^T v_i$, $i = 1, \ldots, n$, we can compute the step size α [see (12)] and p'' (the new iterate) in $O(n)$ time.

5 All vertex triangle algorithm (AVTA)

Given $S = \{v_i \in \mathbb{R}^m : i = 1, \ldots, n\}$, let R be its diameter, i.e. $R = \max\{d(v_i, v_j), v_i, v_j \in S\}$. Denote the set of vertices of $conv(S)$ by

$$\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}. \tag{15}$$

A straightforward but naive way to compute $\overline{S}$ is to test for each v_i if it lies in $conv(S\backslash\{v_i\})$, to within an ε precision. For each v_i this is an LP-feasibility problem. As usual, given rational inputs, there is an ε_0 so that any $\varepsilon \leq \varepsilon_0$ gives desired precision. Thus it suffices to call n times the convex hull membership oracle. However, this is inefficient. In what follows we describe AVTA which leverages on Triangle Algorithm as membership oracle. The advantage of Triangle Algorithm is that its *Distance Duality* allows AVTA to capture extreme points strategically which leads to a more efficient complexity than the straightforward algorithm.

Definition 4 We say $conv(S)$ is Γ_*-*robust* if

$$\Gamma_* = \min\{d(\overline{v}_i, conv(\overline{S}\backslash\{\overline{v}_i\})) : i = 1, \ldots, K\}. \tag{16}$$

As an example, given a triangle with vertices v_1, v_2, v_3, Γ_* is the minimum of the distances from each vertex to the line segment determined by the other vertices. Thus if other points are placed inside the triangle Γ_* will not be affected.

The following is immediate from Definition 4.

Proposition 2 *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of $\overline{S}$. Suppose $conv(S)$ is Γ_*-robust. Given $v \in S\backslash\widehat{S}$, if for some $\gamma \leq \gamma_* \equiv \Gamma_*/R$ we have $d(v, conv(\widehat{S})) < \gamma R$, then $v \notin \overline{S}$.*

Theorem 3 *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of $\overline{S}$. Given $\gamma \in (0,1)$, consider testing if a given $v \in S\backslash\widehat{S}$ satisfies $d(v, conv(\widehat{S})) \leq \gamma R/2$. Suppose we are given $p' \in conv(\widehat{S})$ for which $\|p'\|^2$ as well as $p'^T\widehat{v}_i$, $i = 1, \ldots, N$ are computed. Then the number of operations to check if $d(v, conv(\widehat{S})) \leq \gamma R/2$ satisfies*

$$O\left(mK^2 + \frac{K}{\gamma^2}\right). \tag{17}$$

Proof Proof is immediate from Theorem 14 and the fact that $N \leq K$. □

We now describe a modification of the Triangle Algorithm for computing all vertices of $conv(S)$. We call this *All Vertex Triangle Algorithm* or simply AVTA. Assume $conv(S)$ is Γ_*-robust, where Γ_* may or may not be available. However, assume we have a constant $\gamma \in (0, 1)$ known to satisfy $\gamma \leq \gamma_* = \Gamma_*/R$. AVTA works as follows. Given a working subset $\widehat{S}$ of $\overline{S}$, initially of cardinality $N = 1$ (see Proposition 3), a single vertex of S, it arbitrarily selects $v \in S\backslash\widehat{S}$. It then tests via the Triangle Algorithm if $d(v, conv(\widehat{S})) \leq \gamma R/2$. If so, it discards v since by definition of γ it cannot belong to $\overline{S}$ (see Proposition 2). Otherwise, it computes a v-witness $p' \in conv(\widehat{S})$. It then sets $c' = v - p'$ and maximizes $c'^T x$ where x ranges in $conv(S\backslash\widehat{S})$. The maximum value coincides with the maximum of $c'^T v_i$ where v_i ranges in $S\backslash\widehat{S}$. If the set of optimal solutions is denoted by S', then $conv(S')$ is a face of $conv(S)$. A vertex v' of $conv(S')$ is a point in S' and is necessarily a vertex of $conv(S)$. Such a vertex can be computed efficiently. Having computed a new vertex v' of $conv(S)$, AVTA replaces $\widehat{S}$ with $\widehat{S} \cup \{v'\}$ and the process is repeated. However, if v coincides with v' AVTA selects a new point in $S\backslash\widehat{S}$. Otherwise, AVTA continues to test if the same v (for which a witness was found) is within a distance of $\gamma R/2$ of the convex hull of the augmented set $\widehat{S}$. Also, as an iterate AVTA uses the same witness p' as a warm up initialization. In doing so each selected $v \in S$ is either determined to be a vertex itself, or it will continue to be tested if it is lies to within a distance of $\gamma R/2$ of the growing set $\widehat{S}$. If within $\gamma R/2$ distance, it will be discarded before AVTA tests another point. We will describe AVTA more precisely. However, we first prove the necessary results. The following Lemma is trivial and we omit the proof:

Lemma 1 *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of $\overline{S}$. For a given $v \in S\backslash\widehat{S}$ suppose $p' \in conv(\widehat{S})$ is a v-witness. Let $c' = v - p'$. Then*

$$\max\{c'^T x : x \in conv(S\backslash\widehat{S})\} = \max\{c'^T v_i : v_i \in S\backslash\widehat{S}\}, \tag{18}$$

thus solving this linear program gives a new vertex.

Corollary 1 *Let $c' = v - p'$ be as in Lemma 1, $p' \in conv(\widehat{S})$ for which $\|p'\|^2$ as well as $p'^T\widehat{v}_i$, $i = 1, \ldots, N$ are computed. Then, $\max\{c'^T x : x \in conv(S\backslash\widehat{S})\}$ can be computed in $O(nK)$ operations.*

Proof Since $N \leq K$, for each i, $c'^T v_i$ can be computed in $O(K)$ operations. □

The next theorem is trivial and we omit the proof.

Theorem 4 *Let S' be the set of optimal solutions of $\max\{c'^T x : x \in S\backslash\widehat{S}\}$. Let $v' \in S'$ be a vertex of $conv(S')$. Then v' is a vertex of $conv(S)$, i.e. $v \in \overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$.*

The following shows computing a single vertex of $conv(S)$ is trivial.

Proposition 3 *Given any v in S, let $Farthest(v, S)$ return a point in S that is farthest from v. Then $Farthest(v, S)$ is a vertex of $conv(S)$, hence a member of $\overline{S}$.*

Proof If $Farthest(v, S)$ is not a vertex of $conv(S)$ it can be written as a convex combination of two other points $v_1, v_2 \in conv(S)$. But then $Farthest(v, S)$ is not a vertex of the triangle formed by v, v_1, v_2, a contradiction.

□

While $Farthest(v, S)$ is a simple procedure to capture a vertex, the set of farthest points of all vertices, in the worst case, could generate at most two vertices. In AVTA, it suffices to start with one vertex using the above procedure which takes $O(nm)$ operations. Next we describe AVTA for computing all vertices of $conv(S)$.

AVTA ($S, \gamma \in (0,1)$)

Step 0. Set $\widehat{S} = \{Farthest(v, S)\}$ for some $v \in S$.
Step 1. Arbitrarily select $v \in S \backslash \widehat{S}$.
Step 2. Call **Triangle Algorithm** $(\widehat{S}, v, \gamma/2)$.
Step 3. If the output p' of Step 2 is a v-witness then Goto Step 4. Otherwise, p' is a $\gamma/2$-approximate solution to v. Set $S \leftarrow S \backslash \{v\}$. If $S = \emptyset$, stop. Otherwise, Goto Step 1.
Step 4. Let $c' = v - p'$.
Compute S', the set of optimal solutions of $\max\{c'^T x : x \in S \backslash \widehat{S}\}$. Arbitrarily select $v' \in S'$. $v' \leftarrow Farthest(v', S')$, $\widehat{S} \leftarrow \widehat{S} \cup \{v'\}$.
Step 5. If $v = v'$, Goto Step 1. Otherwise, Goto Step 2.

Remark 2 Here we make remarks about the steps of AVTA. In Step 0 AVTA selects the first vertex. In Step 1 it randomly select a v in $S \backslash \widehat{S}$. In Step 2 AVTA checks if the point v selected in Step 1 is sufficiently close to the convex hull of the current set of vertices, $\widehat{S}$. If so, in Step 3 v is discarded from further considerations. Otherwise, a v-witness p' is at hand. Step 4 then uses this witness to compute a direction, $c' = v - p'$, where the maximization of $c'^T x$ gives a subset S' of S consisting of the optimal solutions. Then a vertex $conv(S')$ will necessarily be a vertex of $conv(S)$. A vertex of $conv(S')$ is selected by choosing an arbitrary $v' \in S'$ and computing its farthest point in S'. It maybe the case that the vertex v' found in Step 4 coincides with v. Step 5 checks if $v' = v$ in which case it select a new v in the updated $S \backslash \widehat{S}$ in Step 1 for consideration. Otherwise, when this new vertex v' is not v itself, in Step 5 in AVTA v is sent back to Step 2 to be reexamined if v is within $\gamma R/2$ distance of the convex hull of augmented $\widehat{S}$.

Example 1 We consider an example of AVTA, see Fig. 2. In this example $S = \{v_1, \ldots, v_{11}\}$. Note that the set of vertices is $\overline{S} = \{v_4, v_{10}, v_6, v_1, v_9, v_2, v_5, v_8\}$. Suppose the current working subset of vertices $\overline{S}$ consists of $\widehat{S} = \{v_1, v_9, v_2, v_5\}$ and $v = v_3$ is randomly selected to be tested if it lies in $conv(\widehat{S})$. A witness $p' \in conv(\widehat{S})$ is computed and with $c' = p' - v$ maximum of $c'^T x$ over $conv(\overline{S} \backslash \widehat{S})$ is attained at $S' = \{v_4, v_7, v_{10}\}$. Subsequently one of the two points v_4 or v_{10} will become the next vertex to be placed in $\widehat{S}$.

The following theorem is one of the main results:

Theorem 5 *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Let R be the diameter of S. Let $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ be the set of vertices of $conv(S)$.*

(1) *Given any prescribed $t \in (0,1)$ in*

$$O\left(nK^{(t)}(m + \frac{1}{t^2})\right) \tag{19}$$

operations AVTA computes a subset $\overline{S}^t$ of $\overline{S}$ of size $K^{(t)}$ so that the distance from each p in $conv(S)$ to $conv(\widehat{S})$ is at most tR.

(2) *Furthermore, suppose that $conv(S)$ is Γ_*-robust and let $\gamma_* = \Gamma_*/R$. Given $\gamma \in (0, \gamma_*)$, the arithmetic operations AVTA takes to computes $\overline{S}$ is:*

$$O\left(nK(m + \frac{1}{\gamma^2})\right). \tag{20}$$

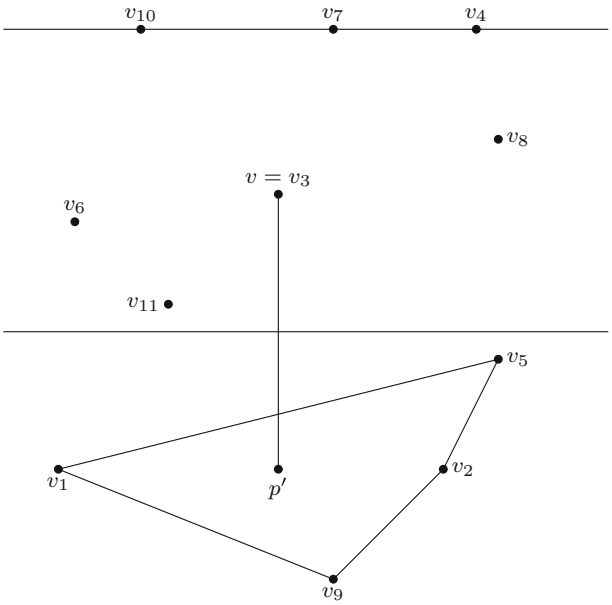

Fig. 2 An example where $\widehat{S} = \{v_1, v_9, v_2, v_5\}$, then $v = v_3$ is randomly selected from $S \backslash \widehat{S}$ and is tested if it lies in $conv(\widehat{S})$. A witness p' is found. Then using $c' - v - p'$ the set $S' = \{v_4, v_7, v_{10}\}$ is computed and one of vertices of $conv(S')$, i.e. v_4 or v_{10} is selected for inclusion in $\widehat{S}$

(3) *If only K is known, the complexity of AVTA to compute $\overline{S}$ is*

$$O\left(\left(nK(m + \frac{1}{\gamma_*^2})\right) \log \frac{1}{\gamma_*}\right). \tag{21}$$

Proof (1) For each input $t \in (0, 1)$, AVTA computes a subset $\overline{S}^t$ of $\overline{S}$ with $K^{(t)}$ to approximate $conv(S)$ with tR precision. Initially in AVTA, the subset $\widehat{S}$ consists of a single element of $\overline{S}$. It continues to grow and computes a subset $\overline{S}^t$ of $\overline{S}$ with $K^{(t)}$ elements to ensure $\forall v \in S$ distance between v and $conv(\overline{S})$ is at most tR. Since the number of vertices needed to give a tR approximation is at most $K^{(t)}$, for each $v \in S \backslash \widehat{S}$ the cost of Step 2 in AVTA is $O(mK^{(t)^2} + K^{(t)}/\gamma^2)$. (Theorem 14 in "Appendix"). Here, (ii) in Theorem 2 is applied with the fact that $R/\Delta \leq 1/t$. The needed inner products in Step 2 are $\widehat{v}_i^T \widehat{v}_j$. However, these inner products need to be computed only once and since there are at most $K^{(t)}$ of $\widehat{v}_i$'s, these inner products can be computed at the cost of $O(mK^{(t)^2})$ operations. We can store the values of the inner products in an array. Then we use them again as they arise in subsequent iterations. This kind of storing can be done for other inner products that may need to be computed in the course of the algorithm. When a selected v is within the distance of $t/2$ to $conv(\widehat{S})$, Step 3 eliminates it from further considerations since v is well approximated. If v is not eliminated, it either gives rise to a new vertex $v' \in \overline{S}$, or v is a vertex itself. In either case, in order to identify a new vertex of $\overline{S}$, after a witness has become available, it requires the minimization of $c'^T v_i$ as v_i ranges over current set of vertices, $S \backslash \widehat{S}$. Since $c' = v - p'$, $p' = \sum_{j=1}^{N} \alpha_j \widehat{v}_j$, where $N = |\widehat{S}|$, the evaluation of $c'^T v_i$ requires the computation of $v^T v_i$, and $v_i^T \widehat{v}_j$, $j = 1, \ldots, N$. This requires $O(nm)$ operations. Since such computation is only required of each vertex in $\overline{S}$, over all the computation of all $c'^T v_i$ requires

$O((n-K^{(t)})mK^{(t)}) = O(nmK^{(t)})$ operations. These together with Theorem 14 imply that the over all complexity is $O(mK^{(t)^2}+nmK^{(t)}+nK^{(t)}/\gamma^2)$ which is the claimed complexities in (1). Next we prove for each $p \in conv(S)$, the distance from p to $conv(\overline{S})$ is at most tR. To begin with, for the 'missing vertices', i.e. $\forall \overline{v}_i \in \overline{S}\backslash\overline{S}^t$, we have $d(v, conv(\overline{S}^t)) \le tR$. For and $p \in conv(S)$, we have:

$$p = \sum_{i=1}^{K} \alpha_i \overline{v}_i, \quad \sum_{i=1}^{K} \alpha_i = 1, \quad \alpha_i \ge 0. \tag{22}$$

Thus

$$d(p, conv(\overline{S}^t)) \le \sum_{i=1}^{K} \alpha_i d(\overline{v}_i, conv(S^t)) \le \sum_{i=1}^{K} \alpha_i tR = tR \tag{23}$$

(2) Since a value $\gamma \le \gamma_*$ is given, it suffices to apply proof of (1) with $t = \gamma$. Note that in each time Step 3 eliminates v, v can not be a vertex as v is $\gamma R/2$ close to $conv(\widehat{S})$. By the definition of γ_*, v can not be a vertex.

(3) When only K is known, we execute AVTA multiple times with different γ which is initialized at 0.5. If we compute K vertices with this estimate of $\gamma_* = \Gamma_*/R$, we stop. Otherwise, we halve γ and repeat the process. Eventually in $O(\log(\gamma_*^{-1}))$ calls to AVTA we accumulate all K vertices in $\overline{S}$. Note each call of AVTA takes $O\left(nK(m+\frac{1}{\gamma_*^2})\right)$ operations. □

Remark 3 If neither K nor an estimate γ to $\gamma_* = \Gamma_*/R$ are known, initially we select $t = 0.5$ and with this value of t compute a subset of vertices with $K^{(t)}$ elements. We can then halve t and repeat the process. Intuitively, if for two consecutive values of t no more vertices are generated we can terminate the process, or decrease t by a factor of four. If Γ_* is not too small we will produce a reasonably good subset of $\overline{S}$ within a reasonable number of calls to AVTA. In either case we are assured of an approximation of $conv(S)$ according to (1) in Theorem 5.

5.1 Application of AVTA in solving the convex hull membership

Suppose we wish to solve the convex hull membership problem: Test if a particular point p lies in $conv(S)$, $S = \{v_1, \ldots, v_n\}$. This is equivalent to linear programming and thus can be solved with variety of algorithms, including polynomial-time algorithms, the simplex method, Frank–Wolfe, or triangle Algorithm. Whichever algorithm we use, the number n plays a role in the complexity. Thus if we compute the set of vertices of $conv(S)$, $\overline{S}$, we can then test if p lies in $conv(\overline{S})$ with K instead of n. This approach may seem to be inefficient, however depending upon the accuracy to which we wish to solve the problem and the size of γ_* it may result in a more efficient algorithm. The next theorem considers the application of Theorem 5 in solving the convex hull membership problem.

Theorem 6 *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Let R be the diameter of S. Let $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ be the set of vertices of $conv(S)$. Suppose $conv(S)$ is Γ_*-robust. Given any $0 < \gamma \le \Gamma_*/R$, the number of operations to test if for a given $p \in \mathbb{R}^m$ admits an ε-approximate solution is*

$$O\left(nmK + \frac{nK}{\gamma^2} + \frac{K}{\varepsilon^2}\right). \tag{24}$$

Proof To test if p admits an ε-approximate solution can be achieved by first computing the vertices in S, followed by testing if p admits an ε-approximate solution in $conv(S)$. From Theorems 2 and 4 it follows that the total complexity is as claimed. □

Remark 4 It is easy to check that for some values of $\varepsilon < \gamma$ the computations of $\overline{S}$ followed by testing if p lies in $conv(\overline{S})$ could be more efficient than solving the convex hull membership without computing $\overline{S}$. This is especially true when $K = o(n)$.

6 AVTA under input perturbation

As in the previous section, we assume $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, R the diameter of S, and $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ the set of vertices of $conv(S)$. Assume $conv(S)$ is Γ_*-robust.

As before we wish to compute $\overline{S}$ or a reasonable subset of it. However, in practice the input set S may be not S but a perturbation of S. This changes the set of vertices, robustness parameter and more. We wish to study perturbations under which we can recover the corresponding perturbation of $\overline{S}$ and extend AVTA to computing this perturbation.

Definition 5 For a given $\varepsilon \in (0, 1)$ the ε-perturbations of S is the set S_ε defined as

$$S_\varepsilon = \{v_1^\varepsilon, \ldots, v_n^\varepsilon\}, \quad \|v_i - v_i^\varepsilon\| \leq \varepsilon R. \tag{25}$$

The ε-perturbations of $\overline{S}$ is the set $\overline{S}_\varepsilon$, denoted by

$$\overline{S}_\varepsilon = \{\overline{v}_1^\varepsilon, \ldots, \overline{v}_K^\varepsilon\}, \tag{26}$$

where $\overline{v}_i^\varepsilon$ is the perturbation of $\overline{v}_i$.

In practice we may be given S_ε as opposed to S. The first question that arises is: What is the relationship between the vertices of S and those of S_ε? Without any assumptions, the vertices of $conv(S_\varepsilon)$ could change drastically, even under small perturbations.

Example 2 Consider a triangle with three additional interior points, very close to its vertices. It may be the case that even under small perturbation all six points become vertices, or that the interior points become the new vertices while the vertices become the new interior points. Thus there is a need to make some assumptions before we can say anything about the nature of perturbed points.

We would hope that for appropriate range of values of ε, $\overline{S}_\varepsilon$ would at least be a subset of the set of vertices of S_ε. First we need a definition.

Definition 6 We say $conv(S)$ is Σ_*-*weakly robust* if

$$\Sigma_* = \min\{d(v, conv(S\backslash\{v\})) : v \in \overline{S}\}. \tag{27}$$

Example 3 Suppose that S consists of the vertices of a non-degenerate triangle with vertices v_1, v_2, v_3. Suppose one additional point is placed inside the triangle. Then clearly $\Sigma_* < \Gamma_*$.

More generally we have

Proposition 4 *Given $S = \{v_1, \ldots, v_n\}$, we have*

$$\Sigma_* \leq \Gamma_*. \tag{28}$$

Other than the inequality in Proposition 4, Σ_* and Γ_* corresponding to the set S may seem unrelated, however in the following theorem we establish a relationship between the two that is useful in the analysis of AVTA for computing $\overline{S}_\varepsilon$.

Theorem 7 *Let S and $\overline{S}$ be as before. Suppose $conv(S)$ is Γ_*-robust, also Σ_*-weakly robust. Let $\rho_* = \min\{d(v_i, v_j) : v_i, v_j \in S, i \neq j\}$. We have*

$$\Sigma_* \geq \frac{\rho_*}{R}\Gamma_* = \rho_*\gamma_*. \tag{29}$$

Proof For each vertex $v \in conv(S)$, let Γ_v be the distance from v to the convex hull of the remaining vertices in S. Specifically,

$$\Gamma_v = d(v, conv(\overline{S}\backslash\{v\})). \tag{30}$$

Also let Σ_v be the distance from v to the convex hull of all other points in S. Specifically,

$$\Sigma_v = d(v, conv(S\backslash\{v\})). \tag{31}$$

Clearly we have,

$$\Sigma_v \leq \Gamma_v. \tag{32}$$

Assume v is a vertex for which $\Sigma_v < \Gamma_v$. If no such a vertex exists then $\Sigma_* = \Gamma_*$ (see Fig. 3). Let u be the closest point to v lying in the convex hull of the the other vertices of S. Thus

$$\Gamma_v = d(v, u), \quad u \in conv(\overline{S}). \tag{33}$$

Let H_u be the hyperplane orthogonal to the line segment vu, passing through u. By definition of u and Carathéodorey's theorem u is a convex combination of vertices of $conv(S)$ lying on H_u. Thus for some subset T of $\overline{S}$ lying on H_u

$$u = \sum_{\overline{v}_i \in T \subset \overline{S}} \alpha_i \overline{v}_i, \quad \sum_{\overline{v}_i \in T \subset \overline{S}} \alpha_i = 1, \quad \alpha_i \geq 0. \tag{34}$$

Figure 3 gives a depiction of this property for a simple example. In the example u is a convex combination of $\overline{v}$ and $\overline{v}'$, vertices of $conv(S)$ lying in the intersection of H_u and $conv(S)$. Consider one of these vertices, say $\overline{v}$. Moving the hyperplane H_u parallel to itself toward v, it intersects the line segment uv at a unique point w that lies on a facet of $conv(S\backslash\{v\})$. Such w exists because $\Sigma_* < \Gamma_*$. In other words, if H_w is a hyperplane parallel to H_u passing through w, then the region of $conv(S)$ enclosed between the halfspace defined by H_w and v contains no point of S in its interior (see shaded area in Fig. 3). This implies

$$\Sigma_v \geq d(v, w). \tag{35}$$

Now consider the intersection of H_w and each ray connecting v to $\overline{v}_i \in T$. Denote this intersection by y_i. In the figure the intersection of H_w and the ray connecting $v\overline{v}$ is denoted by y. By definition of w and Carathéodorey's theorem there must exist a point $v_j \in S$ lying on H_w. Furthermore, v_j can be written as a convex combination of all the y_i's. Thus may may write

$$v_j = \sum_{\overline{v}_i \in T \subset \overline{S}} \beta_i y_i, \quad \sum_{\overline{v}_i \in T \subset \overline{S}} \beta_i = 1, \quad \beta_i \geq 0. \tag{36}$$

Since by definition of ρ_*, $d(v, v_j) \geq \rho_*$, at least for one y_i we must have $d(v, y_i) \geq \rho_*$. This implies we could assume $\overline{v}$ was chosen so that the corresponding y satisfies

$$d(v, y) \geq \rho_*. \tag{37}$$

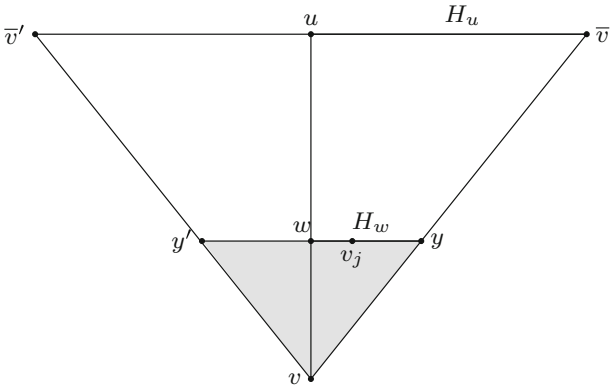

Fig. 3 Given $v \in \overline{S}$, u is its closet point in $conv(\overline{S}\backslash\{v\})$. $\overline{v}, \overline{v}' \in \overline{S}$ are vertices of $conv(S)$ lying on H_u, the orthogonal hyperplane to line segment uv at u. u is a convex combination of these vertices. Moving H_u parallel to itself toward v, it intersects the line segment uv at a unique point w lying on a facet of $conv(S\backslash\{v\})$. Thus interior of shaded region contains no point of S

From similarity of the triangles $\triangle vu\overline{v}$ and $\triangle vwy$ we may write

$$\frac{d(v,w)}{\Gamma_v} = \frac{d(v,y)}{d(v,\overline{v})}. \tag{38}$$

From the definition of R as the diameter of S, $d(v,\overline{v}) \leq R$. From (38), (35) and (37) it follows that

$$\Sigma_v \geq d(v,w) \geq \frac{1}{R} d(v,y)\Gamma_v \geq \frac{1}{R}\rho_*\Gamma_*. \tag{39}$$

This means we have

$$\Sigma_* \geq \frac{1}{R}\rho_*\Gamma_*. \tag{40}$$

□

In what follows we will derive complexity bounds for computing $\overline{S}_\varepsilon$. These complexities will in particular depend on Σ_* or any lower bound σ on $\sigma_* = \Sigma_*/R$. Theorem 6 implies that we can choose $\sigma = \rho_*\Gamma_*/R$.

The following theorem describes a simple condition under which the set of vertices of $conv(S)$ under perturbation remain to be vertices of the perturbed convex hull.

Theorem 8 *Let S be as before, R diameter of S. Suppose $conv(S)$ is Σ_*-weakly robust. Suppose S_ε is an ε-perturbation of S. Let σ be a positive number satisfying $\sigma \leq \sigma_* = \Sigma_*/R$. Assume $\varepsilon < \sigma/2$. If $v \in S$ is a vertex $conv(S)$ and $v^\varepsilon \in S_\varepsilon$ its corresponding ε-perturbation, then v^ε is a vertex of $conv(S_\varepsilon)$.*

Proof Suppose v^ε is not a vertex of $conv(S_\varepsilon)$. Without loss of generality assume $v = v_1$. Hence, $v^\varepsilon = v_1^\varepsilon$. Thus $v^\varepsilon \in conv(S_\varepsilon\backslash\{v^\varepsilon\})$. We may write

$$v^\varepsilon = \sum_{i=2}^{n} \alpha_i v_i^\varepsilon, \quad \sum_{i=2}^{n} \alpha_i = 1, \quad \alpha_i \geq 0. \tag{41}$$

Set

$$u = \sum_{i=2}^{n} \alpha_i v_i. \tag{42}$$

On the one hand we have

$$u - v^{\varepsilon} = \sum_{i=2}^{n} \alpha_i (v_i - v_i^{\varepsilon}). \tag{43}$$

Then by the triangle inequality

$$\|u - v^{\varepsilon}\| \leq \sum_{i=2}^{n} \alpha_i \|v_i - v_i^{\varepsilon}\| \leq \sum_{i=2}^{n} \alpha_i \varepsilon R = \varepsilon R. \tag{44}$$

On the other hand, v is in $\overline{S}$. Without loss of generality assume $v = \overline{v}_1$. From this assumption and since by (42) $u \in conv(\overline{S} \backslash \{\overline{v}_1\})$ we have

$$u = \sum_{i=2}^{K} \gamma_i \overline{v}_i, \quad \sum_{i=2}^{K} \gamma_i = 1, \quad \gamma_i \geq 0. \tag{45}$$

Since $conv(S)$ is Σ_*-weakly robust on $\overline{S}$ and $\sigma \leq \sigma_* = \Sigma_*/R$ we have,

$$\|u - v\| \geq \sigma R. \tag{46}$$

However, from (44), the fact that $\|v - v^{\varepsilon}\| \leq \varepsilon R$ and the triangle inequality we may write.

$$\|u - v\| = \|u - v^{\varepsilon} + v^{\varepsilon} - v\| \leq \|u - v^{\varepsilon}\| + \|v^{\varepsilon} - v\| \leq \varepsilon R + \varepsilon R = 2\varepsilon R. \tag{47}$$

This contradicts the assumption that $2\varepsilon < \sigma$. Hence v^{ε} is a vertex of $conv(S_{\varepsilon})$. □

Remark 5 The theorem implies that if the input to AVTA is S_{ε} instead of S, AVTA will still return at least K vertices. However, the set of vertices of $conv(S_{\varepsilon})$ may have more elements than K, possibly all of S_{ε}. Moreover, the weakly robustness parameter Σ_* will change. We thus need to revise AVTA if we wish to extract the subset $\overline{S}_{\varepsilon} = \{\overline{v}_1^{\varepsilon}, \ldots, \overline{v}_K^{\varepsilon}\}$ from the set of vertices of $conv(S_{\varepsilon})$.

In what follows we will first show how under a mild assumptions on the relationship between Σ_*/R and ε, AVTA can compute a subset $\widehat{S}_{\varepsilon}$ of the vertices of $conv(S_{\varepsilon})$ containing $\overline{S}_{\varepsilon}$ (Theorem 9). We then show how AVTA can efficiently extract from $\widehat{S}_{\varepsilon}$ the desired set, namely $\overline{S}_{\varepsilon}$. The next lemma establishes a lower bound on the week-robustness of $conv(S_{\varepsilon})$. It also shows how spurious vertices of $conv(S_{\varepsilon})$ are situated with respect to the convex hull of the remaining vertices. This will be used in Theorem 9 in pruning such vertices.

Lemma 2 *Suppose $conv(S)$ is Σ_*-weakly robust. Suppose $\varepsilon < \Sigma_*/2R$. Let v^{ε} be any point in $\overline{S}_{\varepsilon}$. Let $\widehat{S}_{\varepsilon}$) be any subset of vertices of $conv(S_{\varepsilon})$ containing $\overline{S}_{\varepsilon}$). Then,*

$$d(v^{\varepsilon}, conv(\widehat{S}_{\varepsilon})) \geq (\Sigma_* - 2\varepsilon R). \tag{48}$$

Moreover let $\widehat{v}^{\varepsilon}$ be any (spurious) point in $\widehat{S}_{\varepsilon} \backslash \overline{S}_{\varepsilon}$. Then

$$d(\widehat{v}^{\varepsilon}, conv(\widehat{S}_{\varepsilon} \backslash \{\widehat{v}^{\varepsilon}\})) \leq \varepsilon R. \tag{49}$$

Proof By Theorem 8, $\overline{S}_{\varepsilon}$ is a subset of vertices of $conv(S_{\varepsilon})$. Given $v^{\varepsilon} \in \overline{S}_{\varepsilon}$, let v be the corresponding vertex in $\overline{S}$. Given w^{ε} in $conv(S_{\varepsilon} \backslash \{v^{\varepsilon}\})$, let w in $conv(S \backslash \{v\})$ be the corresponding point, i.e. defined with respect to the same convex combination of corresponding vertices. Then

$$\|v - v^{\varepsilon}\| \leq \varepsilon R, \quad \|w - w^{\varepsilon}\| \leq \varepsilon R. \tag{50}$$

From the above it is easy to show

$$|d(v, w) - d(v^\varepsilon, w^\varepsilon)| \leq 2\varepsilon R. \tag{51}$$

But this implies

$$d(v, w) - d(v^\varepsilon, w^\varepsilon) \leq 2\varepsilon R. \tag{52}$$

Equivalently,

$$d(v, w) - 2\varepsilon R \leq d(v^\varepsilon, w^\varepsilon). \tag{53}$$

But $d(v, w) \geq \sigma_* R = \Sigma_*$. This proves (48).

To prove (49), let $\widehat{v}$ be the point in S corresponding to $\widehat{v}^\varepsilon$. We have

$$\widehat{v} = \sum_{i=1}^{K} \alpha_i \overline{v}_i, \quad \sum_{i=1}^{K} \alpha_i = 1, \quad \alpha_i \geq 0. \tag{54}$$

Define

$$\widehat{w} = \sum_{i=1}^{K} \alpha_i \overline{v}_i^\varepsilon, \quad \sum_{i=1}^{K} \alpha_i = 1, \quad \alpha_i \geq 0. \tag{55}$$

It is now easy to show

$$\|\widehat{v}^\varepsilon - \widehat{w}\| \leq \|\widehat{v}^\varepsilon - \widehat{v}\| + \|\widehat{v} - \widehat{w}\| \leq 2\varepsilon R. \tag{56}$$

This proves (49). □

Theorem 9 *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Assume $conv(S)$ is Σ_*-weakly robust. Suppose $\varepsilon \leq \Sigma_*/4R$.*

(i) *Given any $t \in (0, 1)$, AVTA can be modified to compute a subset $\overline{S}_\varepsilon^t$ of the set of vertices of $conv(S_\varepsilon)$ of cardinality $K_\varepsilon^{(t)}$ so that the distance from each point in $conv(S_\varepsilon)$ to $conv(\overline{S}_\varepsilon^t)$ is at most t. In particular, the distance from each point in $conv(S)$ to $conv(S_\varepsilon^t)$ is at most $(t + \varepsilon)R$. The complexity of the computation of $\overline{S}_\varepsilon^t$ is*

$$O\left(nK_\varepsilon^{(t)}\left(m + \frac{1}{t^2}\right)\right). \tag{57}$$

(ii) *Given σ satisfying, $4\varepsilon \leq \sigma \leq \sigma_* = \Sigma_*/R$, AVTA can be modified to compute a subset $\widehat{S}_\varepsilon$ of the set of vertices of S_ε containing $\overline{S}_\varepsilon$, then compute from this subset $\overline{S}_\varepsilon$ itself. If K_ε is the cardinality of $\widehat{S}_\varepsilon$, the total number of operations satisfies*

$$O\left(nK_\varepsilon\left(m + \frac{1}{\sigma^2}\right)\right). \tag{58}$$

(iii) *Given γ, satisfying $4\varepsilon \leq \gamma\rho_* \leq \Gamma_*\rho_*/R = \gamma_*\rho_*$, AVTA can be modified to compute a subset $\widehat{S}_\varepsilon$ of the set of vertices of S_ε containing $\overline{S}_\varepsilon$, then compute from this subset $\overline{S}_\varepsilon$ itself. If K_ε is the cardinality of $\widehat{S}_\varepsilon$ the total number of operations satisfies*

$$O\left(nK_\varepsilon\left(m + \frac{1}{(\rho_*\gamma)^2}\right)\right). \tag{59}$$

(iv) *Given only K, where $4\varepsilon \leq \Sigma_*/R$, the number of operations of AVTA to computes $\overline{S}_\varepsilon$ is.*

$$O\left(nK_\varepsilon\left(m + \frac{1}{\sigma_*^2}\right)\right)\log\left(\frac{1}{\sigma_*}\right). \tag{60}$$

Proof By Theorem 8, $\overline{S}_\varepsilon$ is a subset of vertices of $conv(S_\varepsilon)$. Let $\sigma_\circ = (\Sigma_* - 2\varepsilon R)/R$. Then since $\varepsilon \leq \Sigma_*/4R$, $\sigma_\circ \geq \Sigma_*/2R$. Then by Lemma 2, for each $v^\varepsilon \in \overline{S}_\varepsilon$, we have

$$d(v^\varepsilon, conv(S^\varepsilon \backslash \{v^\varepsilon\}) \geq \Sigma_*/2R. \tag{61}$$

Now consider a modification of AVTA that replaces $\gamma/2$, by $\sigma/2$. Such modified AVTA will compute a subset $\overline{S}_\varepsilon$ of vertices of $conv(S_\varepsilon)$ that must necessarily contain $\overline{S}_\varepsilon$. Analogous to Theorem 5, (2), the complexity of this part is as stated in part (ii) of the present theorem.

Now consider $conv(\widehat{S}_\varepsilon)$ and assume v^ε is a vertex of it within a distance of less than $\sigma/2$, say $\sigma/4$. Then by Lemma 2, $v^\varepsilon \notin \overline{S}^\varepsilon$. We can thus apply the Triangle Algorithm to remove any vertex of $conv(\widehat{S}_\varepsilon)$ that is within a distance of less than $\sigma/2$ of the convex hull of the other vertices in $conv(\widehat{S}_\varepsilon)$. Again analogous to Theorem 5 the over all complexity of this step is bounded by

$$O\left(mK_\varepsilon^2 + \frac{K_\varepsilon^2}{\sigma_\circ^2}\right) = O\left(mK_\varepsilon^2 + \frac{K_\varepsilon^2}{\sigma^2}\right). \tag{62}$$

This is dominated by the complexity of the first part. This proves (i). Proof of (ii) follows from Theorem 7, (29), that $\gamma\rho_* \leq \sigma_*$.

To prove (iv), we start by $\sigma = 1/2$ and run AVTA. Then as previous case prune unwanted vertices. If we end up with $\overline{S}_\varepsilon$, we are done. If not, we repeat the process with $\sigma = 1/4$ and so on. Eventually we will recover $\overline{S}_\varepsilon$.

The proof of (i) is analogous to the proof of Theorem 8, part (3). □

Remark 6 Ideally, K_ε is within a constant multiple of K, in which case the complexities are analogous to those of Theorem 5. In the worst-case $\widehat{S}_\varepsilon = S_\varepsilon$, i.e. $K_\varepsilon = n$. On the other hand, ignoring the size of K_ε, suppose $\sigma_\circ \geq (\sqrt{n/K})\varepsilon$, then the complexity of generating the vertices of $conv(S_\varepsilon)$ is

$$O\left(nmK_\varepsilon + \frac{nK}{\varepsilon^2}\right). \tag{63}$$

7 Triangle Algorithm with Johnson–Lindenstrauss projections

Consider again $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. We wish to compute the subset $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ of all vertices of $conv(S)$. Johnson–Lindenstrauss lemma (Johnson and Lindenstrauss 1984) allows embedding the n points of S in an m'-dimensional Euclidean space, where $\mathbb{R}^{m'}$, $m' < m$, via a randomized linear map so that the distances between every pair of points in S and those of their images in $\mathbb{R}^{m'}$ remain approximately the same, with high probability. More specifically, there is a universal constant c such if ε' satisfies,

$$\frac{c \log n}{m} \leq \varepsilon'^2 < 1, \tag{64}$$

and $m' < m$ is an integer satisfying

$$m' \approx \frac{c \log n}{\varepsilon'^2}, \tag{65}$$

then there exists a randomized linear map $L : \mathbb{R}^m \to \mathbb{R}^{m'}$ so that if $u_i = L(v_i)$, and

$$U = L(S) = \{u_1, \ldots, u_n\} \subset \mathbb{R}^{m'}, \tag{66}$$

then for for each $i, j \in \{1, \ldots, n\}$ we have

$$\Pr\left(d(v_i, v_j)(1-\varepsilon') \le d(u_i, u_j) \le d(v_i, v_j)(1+\varepsilon')\right) > 1 - \frac{2}{n}. \tag{67}$$

The projection of each point takes $O(m \log n)$ operations so that the overall number of operations to project all the n points is

$$O(nm \log n). \tag{68}$$

In this section we consider computing $\overline{S}$, the set of vertices of $conv(S)$ by using the Johnson–Lindenstrauss projections and then computing the set of vertices of $conv(U)$ via AVTA. Let $\overline{U}$ denote the set of vertices of $conv(U)$ and let its cardinality be K'. First we state some properties of $conv(U)$.

Lemma 3 *Given $v \in S$, $L : \mathbb{R}^m \to \mathbb{R}^{m'}$, a randomized linear map, suppose $u = L(v)$ is a vertex of $conv(U)$. Then v is a vertex of $conv(S)$.*

Proof Suppose v is not a vertex of $conv(S)$. Then $v = \sum_{i=1}^n \alpha_i v_i$, $\sum_{i=1}^n \alpha_i = 1$, $\alpha_i \ge 0$, $i = 1, \ldots, n$, with some $0 < \alpha_j < 1$. By linearity of L we have

$$u = L(v) = \sum_{i=1}^n \alpha_i L(v_i) = \sum_{i=1}^n \alpha_i u_i. \tag{69}$$

This implies u is not a vertex of $conv(U)$, a contradiction. □

The next theorem gives an estimate of the robustness parameters of $conv(U)$ in terms of those $conv(S)$.

Theorem 10 *Suppose $conv(S)$ is Γ_*-robust and Σ_*-weakly robust. Let $U = L(S)$, L a randomized linear map, $L : \mathbb{R}^m \to \mathbb{R}^{m'}$. Let m' and ε' be related as in (65). If $conv(U)$ is Γ'_*-robust, Σ'_*-weakly robust, then with probability at least $(1 - 2/n)$, we have*

$$\Gamma'_* \ge \Gamma_*(1-\varepsilon'), \quad \Sigma'_* \ge \Sigma_*(1-\varepsilon'). \tag{70}$$

Proof Suppose u is a vertex of $conv(U)$ and $\widehat{U}$ a subset of its vertices not containing u. Let v and $\widehat{S}$ be the preimages of u and $\widehat{U}$ under the linear map L. By Lemma 3 v and the elements of $\widehat{S}$ are all vertices of $conv(S)$. From (67) it is easy to argue that with probability at least $(1 - 2/n)$ we have

$$d(u, conv(\widehat{U})) \ge d(v, conv(\widehat{S}))(1-\varepsilon'). \tag{71}$$

The claimed inequalities follow. □

From Theorem 10 and Theorem 5 we can state the following:

Theorem 11 *Given $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$ let $U = L(S) = \{u_1, \ldots, u_n\} \subset \mathbb{R}^{m'}$, L a randomized linear map, m', ε' as before. Let $\overline{U} = \{\overline{u}_1, \ldots, \overline{u}_{K_{\varepsilon'}}\}$ be the set of vertices of $conv(U)$. Suppose $conv(S)$ is Γ_*-robust and $conv(U)$ is Γ'_*-robust. Then with probability at least $(1 - 2/n)$,*

(1) *The number of arithmetic operations of AVTA to compute $\overline{U}$ is*

$$O\left(nK_{\varepsilon'}(m' + \frac{nK_{\varepsilon'}R^2}{\Gamma'^2_*})\right) = O\left(\frac{n \log nK_{\varepsilon'}}{\varepsilon'^2} + \frac{nK_{\varepsilon'}}{\gamma_*^2(1-\varepsilon')^2}\right). \tag{72}$$

(2) *Given any prescribed positive* $t \in (0, 1)$*, AVTA in*

$$O\left(nK^t_{\varepsilon'}(m' + \frac{1}{t^2})\right) = O\left(\frac{n \log nK^t_{\varepsilon'}}{\varepsilon'^2} + \frac{nK^t_{\varepsilon'}}{t^2}\right) \tag{73}$$

operations can compute a subset $\overline{U}^t$ *of* $\overline{U}$ *of size* $K^t_{\varepsilon'}$ *so that the distance from each point in* $conv(U)$ *to* $conv(\overline{U}^t)$ *is at most* t*.* □

Remark 7 The results in this section and the above theorem suggest a heuristic approach as an alternative to using AVTA directly to compute all the vertices of $conv(S)$: Compute $U = L(S)$, the Johnson–Lindenstrauss projection of S under a randomized linear map L. Then apply AVTA to compute all the vertices of $conv(U)$, $\overline{U}$. This identifies $|\overline{U}| \leq K$ vertices of $conv(S)$. Next move up to the full dimension and continue with AVTA to recover the remaining vertices of $conv(S)$. Alternatively, we can repeat randomized projections and compute the corresponding vertices. We would have to delete duplications which is not difficult, given that we store the computed vertices via their vector of representation of convex combination coefficients. We would expect that when sufficient number of projections are applied all vertices of $conv(S)$ can be recovered. However, in the remaining of the section we analyze the probability that under a random projection, the projection of a vertex of $conv(S)$ is a vertex of the projection.

In what follows we first state a result on Johnson–Lindenstrauss random projections on the convex hull membership problem from Vu et al. (2017). Next we state an alternative result.

Proposition 5 (Vu et al. 2017, Proposition 3.3) *Given* $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$*,* $p \in \mathbb{R}^m$ *such that* $p \notin conv(S)$*, let* $d = \min\{d(p, x) : x \in conv(S)\}$ *and* $D = \max\{d(p, v_i) : i = 1, \ldots, n\}$*. Let* $T : \mathbb{R}^m \to \mathbb{R}^k$ *be a random linear map. Then*

$$\text{Prob}\Big(T(p) \notin T(conv(S))\Big) \geq 1 - 2n^2 e^{-c(\varepsilon^2 - \varepsilon^3)k} \tag{74}$$

for some constant c *(independent of* m, n, k, d, D*) and* $\varepsilon < d^2/D^2$*.*

Remark 8 Note that $k = O(\ln n/\epsilon^2) = O(\ln nD^4/d^4)$.

The following is an alternative to Proposition 5 based on the Distance Duality theorem (1) and generally gives a better estimate of ε, hence a smaller k than Proposition 5.

Theorem 12 *Given* $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$*,* $p \in \mathbb{R}^m$ *such that* $p \notin conv(S)$*, let* $d = \min\{d(p, x) : x \in conv(S)\}$*,* $p_* = \text{argmin}\{d(p, x) : x \in conv(S)\}$ *and* $D = \max\{d(p, v_i) : i = 1, \ldots, n\}$*. Let*

$$E = \min\left\{\frac{d(p, v_i)}{d(p_*, v_i)} : i = 1, \ldots, n\right\}. \tag{75}$$

Let $T : \mathbb{R}^m \to \mathbb{R}^k$ *be a random linear map. Then*

$$\text{Prob}\Big(T(p) \notin T(conv(S))\Big) \geq 1 - 2n^2 e^{-c\varepsilon^2 k}, \tag{76}$$

for some constant c *(independent of* m, n, k, d, D*) and* $\varepsilon < (E - 1)/(E + 1)$*. Furthermore,* $(E - 1)/(E + 1) > d^2/4D^2$*.*

Proof Since p_* is the closest point to p in $conv(S)$, it is easy to show that it is a p-witness, i.e.

$$d(p_*, v_i) < d(p, v_i), \quad \forall i = 1, \ldots, n. \tag{77}$$

Let $\overline{p} = T(p)$, $\overline{p}_* = T(p_*)$, and for $i = 1, \ldots, n$, $\overline{v}_i = T(v_i)$. We now consider the set of $n+1$ points $\{v_0 = p, v_1, \ldots, v_n\}$ and their random projections and find condition on ε such that $\overline{p}_*$ will be an $\overline{p}$-pivot with respect to $T(conv(S))$, probabilistically. By the Johnson–Lindenstrauss Lemma we have,

$$\text{Prob}\Big((1-\varepsilon)d(v_i, v_j) \le d(\overline{v}_i, \overline{v}_j) \le (1+\varepsilon)d(v_i, v_j)\Big) \ge 1 - 2(n+1)^2 e^{-c\varepsilon^2 k}, \tag{78}$$

for some constant c (independent of m, n, k). From (78) and definition of E, for each $i = 1, \ldots, n$ with probability at least $1 - 2(n+1)^2 e^{-c\varepsilon^2 k}$ we have,

$$d(\overline{p}_*, \overline{v}_i) \le (1+\varepsilon)d(p_*, v_i) \le \frac{(1+\varepsilon)}{E} d(p, v_i) \le \frac{(1+\varepsilon)}{(1-\varepsilon)} \frac{1}{E} d(\overline{p}, \overline{v}_i). \tag{79}$$

Note that assuming $n \ge 2$, $1 < E < \infty$. We thus restrict ε to satisfy

$$\frac{(1+\varepsilon)}{(1-\varepsilon)} \frac{1}{E} < 1. \tag{80}$$

Equivalently,

$$\varepsilon < \frac{E-1}{E+1}. \tag{81}$$

Thus with ε satisfying the above, $\overline{p}_*$ is a witness with high probability.

Next we find a lower bound on the right-hand-side of the above. Since E is finite, $E = d(p, v_j)/d(p_*, v_j)$ for some j, i.e. $p_* \ne v_j$. Consider the triangle with vertices p, v_j and p_*. With $d(p, p_*)$ and $d(p, v_j)$ fixed, the maximum value of $d(p_*, v_j)$ is $\sqrt{d^2(p, v_i) - d^2(p, p_*)}$. Using this we may write

$$E = \frac{d(p, v_j)}{d(p_*, v_j)} \ge \frac{d(p, v_j)}{\sqrt{d^2(p, v_i) - d^2(p, p_*)}} = \frac{1}{\sqrt{1 - d^2(p, p_*)/d^2(p, v_j)}}. \tag{82}$$

But $d(p, p_*) = d$ and $d(p, v_j) \le D$. Thus

$$E \ge \frac{1}{\sqrt{1 - d^2/D^2}} = \frac{D}{\sqrt{D^2 - d^2}}. \tag{83}$$

The function $(x-1)/(x+1)$ is monotonically increasing. Thus from (84) we have

$$\frac{E-1}{E+1} \ge \frac{D - \sqrt{D^2 - d^2}}{D + \sqrt{D^2 + d^2}} = \frac{d^2}{(D + \sqrt{D^2 - d^2})^2} \ge \frac{d^2}{4D^2}. \tag{84}$$

□

Remark 9 We would expect that $(E-1)/(E+1)$ is generally a larger number than $d^2/4D^2$. Thus Theorem 12 gives generally a better estimate of ε and k than those of Proposition 5. An additional advantage of Theorem 12 is that it shows the applicability of the Triangle Algorithm in solving the convex hull membership problem using random projections.

We now state a corollary of the theorem on computation of all vertices of $conv(S)$.

Corollary 2 *Given $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, suppose $conv(S)$ is Γ_*-robust. Let R be the diameter of S. Suppose v_j is a vertex of $conv(S)$. Let $T : \mathbb{R}^m \to \mathbb{R}^k$ be a random linear map. Then the probability that $T(v_j)$ is a vertex of $T(conv(S))$ is at least $1 - 2n^2 e^{-c\varepsilon^2 k}$, for some constant c (independent of m, n, k) and $\varepsilon < \gamma_*^2/4$.*

Proof We apply the previous theorem with v_j as b and considering the probability that under a random projection of v_j lies in projection of the convex hull of the remaining points. Note that $d(v_j, conv(S\backslash\{v_j\})) \geq \Gamma_*$ and $\max\{d(v_j, v_i) : v_i \in S\backslash\{v_j\} \leq R$. Thus we can replace for b/D in (84) in the previous theorem by $\gamma_* = \Gamma_*/R$. Thus we can write $(E-1)/(E+1) \geq \gamma_*^2/4$. This gives the upper bound on ε. □

7.1 AVTA under perturbation and Johnson–Lindenstrass projection

Let S_ε be as before and U_ε, a subset of $\mathbb{R}^{m'}$ the perturbation of U. Let $\overline{U}_\varepsilon$ be the perturbation of $\overline{U}$. Based on the results in this section and previous complexity bounds we have

Theorem 13 *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Assume $conv(S)$ is Σ_*-weakly robust. Suppose $\varepsilon < \Sigma_*/4R$. Let $\sigma_\circ = (\Sigma_* - 2\varepsilon R)/R = \sigma_* - 2\varepsilon$. Then with probability at least $(1 - 2/n)$,*

(i) *AVTA can be modified to compute a subset $\widehat{U}_\varepsilon$ of U_ε, of cardinality $K_{\varepsilon\varepsilon'}$ such that it contains $\overline{U}_\varepsilon$. Then AVTA can compute from this subset $\overline{U}_\varepsilon$ itself, where the total number of operations satisfies*

$$O\left(nm' K_{\varepsilon\varepsilon'} + \frac{nK_{\varepsilon\varepsilon'}}{\sigma_\circ^2(1-\varepsilon')^2}\right) = O\left(\frac{n \log n K_{\varepsilon\varepsilon'}}{\varepsilon^2} + \frac{nK_{\varepsilon\varepsilon'}}{\sigma_\circ^2(1-\varepsilon')^2}\right). \tag{85}$$

(ii) *Given any prescribed positive $t \in (0, 1)$, in*

$$O\left(\frac{n \log n K_{\varepsilon\varepsilon'}^{(t)}}{\varepsilon^2} + \frac{nK^{(t)}\varepsilon\varepsilon'}{\overline{\sigma}_\circ^2(1-\varepsilon')^2}\right) \tag{86}$$

operations the modified AVTA can compute a subset U_ε^t of $\overline{U}_\varepsilon$ of size $K_{\varepsilon\varepsilon'}^{(t)}$ so that the distance from each point in $conv(U_\varepsilon)$ to $conv(U_\varepsilon^t)$ is at most t.

8 Applications

While the modified AVTA algorithm comes with theoretical guarantees, in certain cases the algorithm might output many more vertices, K_ε, than desired. Here we present a practical implementation that always outputs exactly K vertices, provided K is known. When K is unknown, our experiments in the next section reveal that the algorithm can automatically detect a slightly larger set that contains a good approximation to the K vertices of interest. Notice that we want a fast way to detect good approximations to the original vertices of the set S and prune out spurious points, i.e., additional vertices of the set S_ε. The key insight on top of the AVTA algorithm is the following: *If the perturbed set is randomly projected onto a lower dimensional space, it is more likely for an original vertex to still be a vertex than for a spurious vertex*. Using this insight the algorithm outlined below runs the modified AVTA algorithm over several random projections and outputs the set of points that appear as vertices in many random projections.

AVTA with multiple random projections ($S = \{v_1, \ldots, v_n\}$, K, γ, M)

Step 0. Set $Freq \leftarrow 0^{|S|}$.
Step 1. For $i = 1$ to M:

$S' \leftarrow S$: Project data on to randomly chosen $\frac{4log(n)}{\epsilon^2}$ dimensions.
- $\widehat{S} \leftarrow \text{AVTA}(S', \gamma)$
- For each $d_j \in \widehat{S}$, $Freq[j] = Freq[j] + 1$.

Step 2. Output top K frequent vertices.

We now show how AVTA can be used to solve various problems in computational geometry and machine learning.

Application of AVTA in linear programming Consider linear programming feasibility problem of testing if $P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$ is nonempty, where A is $m \times n$, $b \in \mathbb{R}^n$. Suppose n is much larger than m. If we reduce the size of A the problem would be more efficiently solvable, no matter what algorithm we use to solve it.

Proposition 6 *Given $P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$, let $conv(A)$ denote the convex hull of columns of A. Let A' denote the $m \times n'$ submatrix A whose columns form the set of all vertices of $conv(A)$. Let*

$$P' = \{x' \in \mathbb{R}^{n'} : A'x' = b, x' \geq 0\}. \tag{87}$$

Then P is feasible if and only P' is feasible.

Proof Clearly, if P' is feasible then P is feasible. Assume P is feasible. Thus for some $x \in \mathbb{R}^n$, $x \geq 0$, $Ax = b$. Denote the columns of A by $a^{(i)}$. Then each $a^{(i)}$ is a convex combination of columns of A'. That is, for each $i = 1, \ldots, n$, there exists

$$\alpha^{(i)} \in S_{n'} = \{s \in \mathbb{R}^{n'} : \sum_{i=1}^{n'} s_i = 1, s \geq 0\}, \tag{88}$$

where

$$a^{(i)} = A'\alpha^{(i)}. \tag{89}$$

Thus

$$Ax = \sum_{i=1}^{n} x_i a^{(i)} = \sum_{i=1}^{n} x_i A'\alpha^{(i)} = A' \sum_{i=1}^{n} x_i \alpha^{(i)}. \tag{90}$$

Letting

$$x' = \sum_{i=1}^{n} x_i \alpha^{(i)}, \tag{91}$$

$A'x' = b$, $x' \geq 0$. □

Proposition 7 *Assume $P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$ is nonempty. Consider the linear program $\min\{c^T x : x \in P\}$. Let B be the $(m + 1) \times n$ matrix whose first row is c^T and the remaining rows are A. Let B' be the $(m + 1) \times n'$ matrix whose columns form the vertices of the convex hull of the columns of B. Let c'^T be the first row of B' and A' the remaining $m \times n'$ submatrix of B'. Then*

$$\min\{c^T x : Ax = b, x \geq 0\} = \min\{c'^T x : A'x' = b, x' \geq 0\}. \tag{92}$$

Proof Consider any feasible solution x_0 of original LP. Then by Proposition 6 the set $\{c'^T x' = c^T x_0, A'x' = b, x' \geq 0\}$ is feasible. This implies the original LP has a finite optimal value if and only if the restricted problem does. In particular, the optimal objective values of the two problems coincide. □

The above propositions imply that AVTA has potential applications in the reduction of the LP feasibility or optimization, whether we solve the problem via simplex method or other methods.

AVTA for topic modeling in the presence of anchor words Arora et al. (2013) provide a practical algorithm for topic modeling with provable guarantees. Their algorithm works under the assumptions that the topic-word matrix is *separable*. In particular, they assume that corresponding to each topic i, there exists an *anchor word* w_i that has a non zero probability of appearing only under topic i. Under this assumption, the algorithm of Arora et al. (2013) consists of two stages: a) find the anchor words, and b) use the anchor words to learn the topic word matrix. The problem of finding anchor words corresponds to finding the vertices of the convex hull of the word-word covariance matrix. They propose an algorithm named *fast anchor words* in order to find the vertices. Since AVTA works in general setting, we can instead use AVTA to find the anchor words. Additionally, the fast anchor words algorithm needs to know the value of the number of anchor words, as an input. On the other hand, from the statements of Theorems 5 and 9 it is easy to see that AVTA can work in a variety of settings when other properties of the data are known such as the robustness. We argue that robustness is a parameter that can be tuned in a better manner than trying different values of the number of anchor words. In fact, one can artificially add random noise to the data and make it robust up to certain value. One can then run AVTA with the lower bound on robustness as input and let the algorithm automatically discover the number of anchor words. This is much more desirable in practical settings. Our first implementation of AVTA is named AVTA+RecoverL2 that uses AVTA to detect anchor words and then uses the anchor words to learn the topic word matrix using the approach from Arora et al. (2013). AVTA is also theoretically superior than fast anchor words and achieves slightly better run times in the regime when the number of topics is $o(\log n)$, where n is the number of words in the vocabulary. This is usually the case in most practical scenarios.

AVTA for topic modeling in the absence of anchor words The presence of anchor words is a strong assumption that often does not hold in practice. Recently, the work of Bansal et al. (2014) designed a new practical algorithm for topic models under the presence of catch words. Catch words for topic i correspond to set S_i such that its total probability of appearing under topic i is significantly higher than in any other topic. Their algorithm called TSVD recovers much better reconstruction of the topic-word matrix in terms of the ℓ_1 error. They also assume that for each topic i, there are a few dominant documents that mostly contain words from topic i. The TSVD algorithm works in two stages. In stage 1, the (thresholded) word-document data matrix is projected onto a K-SVD space to compute a different embedding of the documents. Then, the documents are clustered into K clusters. Under the assumptions mentioned above, one can show that the dominant documents for each topic will be clustered correctly. In stage 2, a simple post processing algorithm can approximate the topic-word matrix from the clustering.

We improve on TSVD by asking the following question: *is K-SVD the right representation of the data?*. Our key insight is that if dominant documents are present in the topic, it is easy to show that most other documents will be approximated by a convex combination of the dominant topics. Furthermore, the coefficients in the convex combinations will provide a

much more faithful low dimensional embedding of the data. Using this insight, we propose a new algorithm that runs AVTA on the data matrix to detect vertices and to approximate each point using a convex combination of the vertices. We then use the coefficient matrix as the new representation of the data that needs to be clustered. Once the clustering is obtained, the same post processing step from Bansal et al. (2014) can be used to recover the topic-word matrix. Our results show that the embedding produced by AVTA leads to much better reconstruction error than of that produced by TSVD. Furthermore, K-SVD is an expensive procedure and very sensitive to the presence of outliers in the data. In contrast, our new algorithm called AVTA+CatchWord is much more stable to noise in the data.

AVTA+CatchWord ($S = \{v_1, \ldots, v_n\}$, γ, K, ϵ)

Step 0. Randomly project S onto $2K$ dimensions to get $\widehat{S}$.
Step 1. Compute a a super set of vertices $\bar{V}$ by $AVTA(\widehat{S}, \gamma)$.
Step 2. Prune $\bar{V}$ into $\widehat{V}$ (of size K) by iteratively picking $\bar{v} \in \{\bar{V}\}/\{\widehat{V}\}$ which has the maximum distance to $conv(\widehat{V})$.
Step 3. For each projected point $\widehat{v}_i \in \widehat{S} \backslash \widehat{V}$, compute a vector α_i such that $\|\widehat{V}\alpha_i - \widehat{v}_i\| \leq \epsilon$.
Step 4. Initialize cluster assignment for each point by majority weight: $\underset{j \in [K]}{\operatorname{argmax}} \alpha_j$.
Step 5. Perform clustering using Lloyds algorithm on the embedding provided by the α vectors.
Step 6. Use the post processing as described in Bansal et al. (2014) to recover the topic-word matrix from the clustering.

AVTA for NMF The work of Arora et al. (2012a) showed that convex hull detection can be used to solve the non-negative matrix factorization problem under the separability assumption. We show that by using the more general AVTA algorithm for solving the convex hull problem results in comparable performance guarantee.

9 Applications and experiments[1]

9.1 Feasibility problem

In this section, we present experimental results which empirically show when the problem is 'over complete', AVTA can be a 'shortcut' solution. In another word, given an $m \times n$ matrix A as data, where the convex hull of the columns of A, denoted by $conv(A)$, has K vertices, $K \ll n$. We apply the AVTA to solve 2 classical problems which appear in many applications.

Convex hull membership problem In the experiments, vertices of the convex hull are generated by the Gaussian distribution, i.e. $v_i \sim \mathcal{N}(0, \mathcal{I}_m), i \in [K]$. Having generated the vertices, the 'redundant' points d_j where $d_j \in conv(S), j \in [n - K]$ are produced using random convex combination $d_j = \sum_{i=1}^{K} \alpha_i v_i$. Here α_i are scaled standard uniform random variable where α_i are scaled so that $\sum_{i=1}^{K} \alpha_i = 1$. Specifically, comparison is by fixing $K = 100$, $m = 50$ and n varying from $5000 \sim 500{,}000$. We compare the efficiency

[1] Resources: https://github.com/yikaizhang/AVTA.

Table 2 Running time of convex hull membership (s)

# of redundant pts	AVTA	TA	FW	Simplex
5000	1.75	0.21	0.52	0.9
20,000	1.49	0.66	1.94	2.76
45,000	2.94	1.84	5.51	6.16
80,000	2.71	3.22	10.87	10.63
125,000	3.83	4.28	17.67	15.95
180,000	4.15	5.38	23.14	24.13
245,000	6.95	9.56	33.42	36.96
320,000	8.09	13.24	44.99	44.26
405,000	10.01	14.75	56.35	59.5
500,000	14.12	15.69	70.7	90.41

of 4 algorithms on solving this problem: the Simplex method (Chvatal 1983), the Frank Wolfe Algorithm (FW) (Jaggi 2013), the Triangle Algorithm (TA) (Kalantari 2015), and our algorithm on solving the convex hull membership query problem.

Results on convex hull membership query Table 2 shows when $n \gg K$, AVTA is more efficient than other algorithms solving the convex hull membership problem. This result supports the output sensitivity property of AVTA.

Linear programming feasibility Linear programming feasibility problem is to find a feasible solution of :

$$A\alpha = p, \quad \alpha \geq 0. \tag{93}$$

In another word, to test if $p \in cone(A_j)$ where A_j are columns of A. In case when A is over complete, any feasible p can be represented using only the generators of $cone(A)$ the set $\bar{A} \subset A$. By scaling A so that columns of AD ($D_{ii} = \frac{b}{a \cdot A_i}$) are in a $m-1$ dimensional hyperplane $\langle a, \alpha \rangle = b$, one can find the generators of $cone(A)$ by finding the vertices of the convex hull of the projected points. This could be done efficiently by AVTA. Suppose we have a linear system A and series of query points p , it is sufficient to run AVTA once for dimension reduction and solve the subproblem $\bar{A}\alpha' = p, \alpha' \geq 0$ using simplex method.
We compare the running time of Simplex Method with AVTA+Simplex Method. The generator $\bar{A}$ is entrywise independent $uniform(0, 1)$ random matrix and the 'overcomplete' part of the matrix $\bar{A}^c = A/\bar{A}$ are generated by $\bar{A}^c = \bar{A}B$ where $B \in \mathbb{R}^{K \times (n-K)}$ is entrywise independent $uniform(0, 10)$ random matrix. We set the number of generators $K = 100$, the dimension $m = 50$, and the number of 'redundant' columns $n = 50{,}000$. We simply set half of the query points feasible and rest infeasible. The feasible points p are generated as $p = Ax$ where $x \in \mathbb{R}^n$ is entrywise independent $uniform(0, 1)$ random vector and the infeasible points are generated in the same way as generators.

It can be observed from Fig. 4a and Table 3 that the running time of AVTA+Simplex doesn't have obvious increase while Simplex increases drastically. This suggests the potential applications of AVTA in linear programming feasibility problem.

9.2 Computing all vertices

Compute vertices of convex hull In this section, we compare the efficiency of AVTA with another popular algorithm for finding vertices Quickhull (Barber et al. 1996). We generate

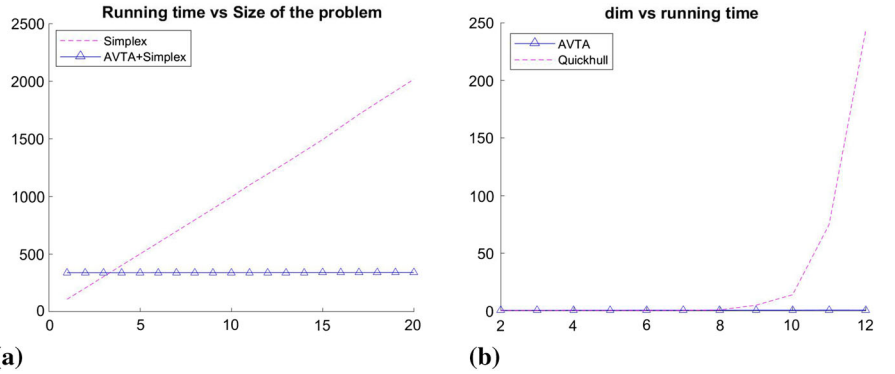

Fig. 4 **a** Running time for algorithms to find a feasible solution **b** Running time for algorithms to find all vertices

Table 3 Running time of linear programming feasibility (s)

# of query	AVTA + simplex	Simplex	# of query	AVTA + simplex	Simplex
1.00	241.24	152.09	11.00	241.94	1810.72
2.00	241.36	303.86	12.00	242.01	1967.93
3.00	241.41	477.89	13.00	242.07	2125.62
4.00	241.45	660.95	14.00	242.16	2289.91
5.00	241.54	853.91	15.00	242.23	2490.52
6.00	241.61	1016.77	16.00	242.29	2680.61
7.00	241.69	1177.30	17.00	242.32	2866.23
8.00	241.72	1336.38	18.00	242.41	3065.50
9.00	241.83	1495.70	19.00	242.44	3245.78
10.00	241.91	1652.84	20.00	242.47	3412.39

vertices according to a Gaussian distribution $\mathcal{N}(0, 10)^m$. Having generated K such points, n interior points are generated as convex combination of the vertices, where the weights are generated scaled i.i.d uniform distribution.

Experiment and results In the experiment, we set $K = 100$, $n = 500$ and m varying from $2 \sim 12$. [2]

The computational results is shown in Table 4. In high dimension $m \geq 9$, when $conv(S)$ is γ robust for some $\gamma > 0$, the AVTA algorithm successfully find all vertices of the convex hull efficiently while the Quick hull algorithm is stuck by its explosion of complexity in dimension m.

Compute vertices of simplex in high dimension The Fast Anchor Word can be used to detect the vertices of a simplex. In this section, we compare the efficiency of AVTA with Fast Anchor Word when convex hull is a simplex with $K = 50$ and $m = 100$. The number of points in the convex hull n varies from 100∼100,000.

[2] The maximum of dimension is 12 in the experiment because of the explosion of running time of the Quick hull algorithm

Table 4 Running time (s)

Dim	Qhull	AVTA	Dim	Qhull	AVTA
2	0.13	14.82	7	2.92	41.51
3	0.02	16.62	8	16.48	39.63
4	0.04	24.49	9	82.09	44.21
5	0.12	32.76	10	391.36	45.79
6	0.59	37.66	11	1479.51	51.19

Results of running time in simplex case The running of efficiency comparison between AVTA and Fast Anchor Word in simplex case is presented in Fig. 5c. In regime $n \geq 30{,}000$, AVTA has less running time.

Compute vertices with perturbation In this section, we compare the robustness of AVTA with multiple random projections presented in Sect. 8 with Fast Anchor Word (Arora et al. 2013). Instead of actual set of points S as input, the algorithm is given a perturbed set $S_\circ$, i.e. S is corrupted by some noise. Having fixed $K = 100, n = 500, m = 100$, we choose a Gaussian perturbation from $\mathcal{N}(0, \tau)^m$ where τ varies from 0.3 to 3. In case of general convex hull, a failure of Fast Anchor Word on computing vertices of general convex hull is presented. The data is generated by setting $\tau = 0.3, m = 50, n = 500$ and let K varies from $10 \sim 100$. We do an error analysis and evaluate the output of the algorithms by measuring the l_2 distance between true vertices and the convex hull of output vertices of the two algorithms. More precisely, given a true vertex $v_i \in S$ and $\widehat{S}$, the output of an algorithm, the error in recovering v_i is defined to be $\min_{u \in conv(\widehat{S})} ||u - v_i||_2$. We add up all the errors to get the total accumulated error.

Results on computing perturbed vertices

The recovery error in robustness comparison is shown in Table 5. The AVTA with multiple random projection has a better recovery error in the simplex case.

It can also be observed from Fig. 5b that in general case, as number of vertices exceeds the number of dimensions, Fast Anchor Word fails to recover more vertices and its error explodes.

9.3 Topic modeling

We compare our algorithms with the Fast Anchor + Recoverl2 algorithm of Arora et al. (2013) and the TSVD algorithm of Bansal et al. (2014) on two types of data sets: semi-synthetic data and real world data. We next describe our methodology and empirical results in detail.

Semi synthetic data For Semi-Synthetic data set, we use similar methodology as in Arora et al. (2013). We first train the model on real data set using Gibbs sampling with 1000 iterations. We choose 50 as the number of topics which follows Bansal et al. (2014). Given the parameters learned from dataset, we generate documents with α set to be 0.01. The average document length is 1000. Then the reconstruction error is measured by the l_1 distance of bipartite matched pairs between the true word-topic distribution and the word-topic distribution (Arora et al. 2013). We then average the errors to compute the final mean error.

Real data We use the **NIPS** data set with 1500 documents , and a pruned vocabulary of 2K words, and the NYTimes Corpus with sub sampled 30,000 documents, and a pruned

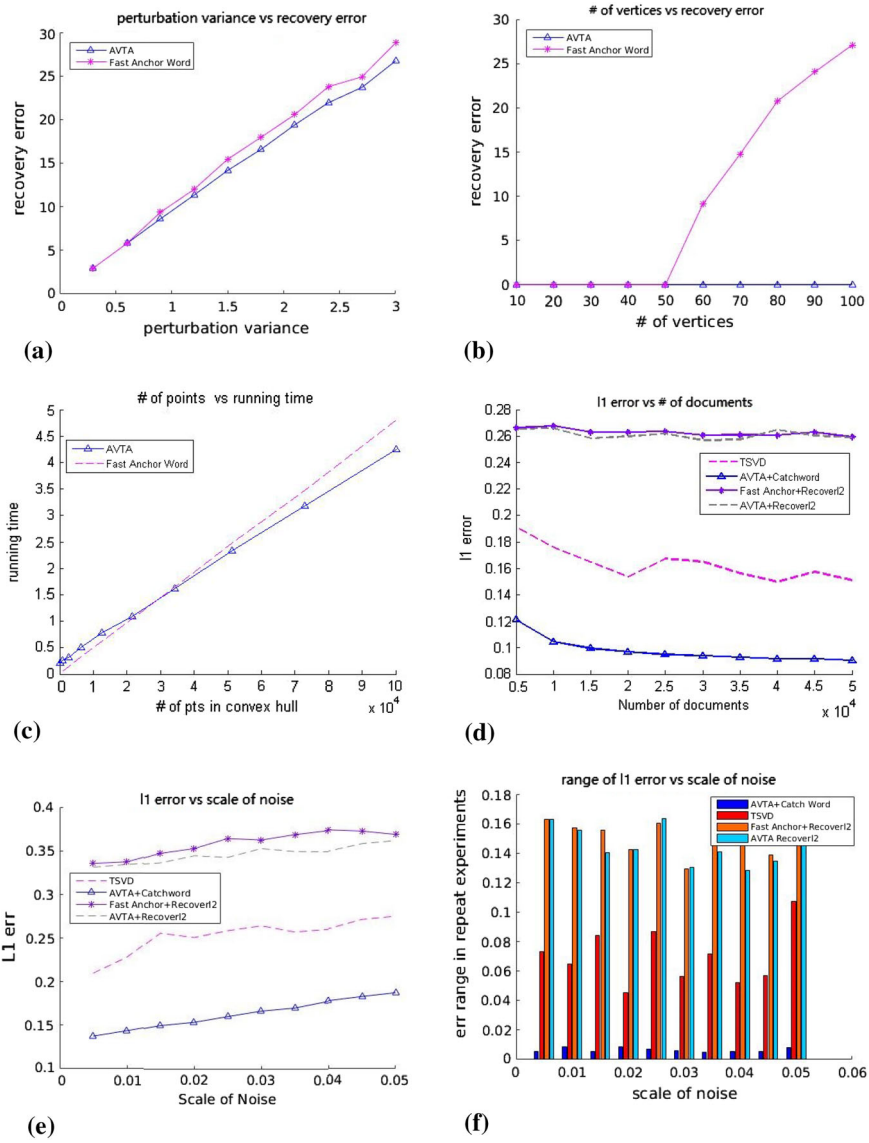

Fig. 5 **a** Recovery error-computing vertices (simplex case) **b** Recovery error-computing vertices (eneral convexhull) **c** Running time (secs) of computing vertices of simplex **d** ℓ_1 error in the semi-synthetic dataset **e** ℓ_1 error in the perturbed semi-synthetic dataset **f** Range of the ℓ_1 error over 10 runs on the noisy semi-synthetic dataset

vocabulary of 5k words.[3] For the real world data set, as in prior works (Arora et al. 2013; Bansal et al. 2014), we evaluate the coherence to measure topic quality (Yao et al. 2009). Given a set of words $\mathcal{W}$ associated with a learned topic, the coherence is computed as: $Coherence(\mathcal{W}) = \sum_{w_1,w_2\in\mathcal{W}} \log \frac{D(w_1,w_2)+\epsilon}{D(w_2)}$, where $D(w_1)$ and $D(w_1, w_2)$ are the number of documents where w_1 appears and (w_1, w_2) appear together respectively (Arora et al.

[3] https://archive.ics.uci.edu/ml/datasets/bag+of+words.

Table 5 Recovery error (simplex)

Var	AVTA + multiple Rp	Fast anchor	Variance	AVTA + multiple Rp	Fast anchor
0.3	2.96	2.96	1.8	16.60	17.98
0.6	5.79	5.79	2.1	19.40	20.58
0.9	8.61	9.36	2.4	21.93	23.77
1.2	11.34	12.00	2.7	23.69	24.90
1.5	14.16	15.44	3	26.72	28.78

2013), and ε is set to 0.01 to avoid w_1, w_2 that never co-occur (Stevens et al. 2012). The total coherence is the sum of the coherence of each topic. In the NIPS dataset, 1000 out of the 1500 documents were selected as the training set to learn the word-topic distributions. The rest of the documents were used as the testing set.

Implementation details We compare 4 algorithms, AVTA + CatchWord, TSVD, the Fast Anchor + Recoverl2 and the AVTA + Recoverl2. We implement our own version of Fast Anchor + Recoverl2 as described in Arora et al. (2013). TSVD is implemented using the code provided by the authors in Bansal et al. (2014). AVTA+Recoverl2 corresponds to using AVTA to detect anchor words from the word-word covariance matrix and then using the Recoverl2 procedure from Arora et al. (2013) to get the topic-word matrix. AVTA + CatchWord corresponds to finding the low dimensional embedding of each document in terms of the coefficient vector of its representation in the convex hull of the vertices. The next step is to cluster these points. In practice, one could use the Lloyd's algorithm for this step which could be sensitive to initialization. To remedy this, we use similar heuristic as Bansal et al. (2014) of the initialization step. We repeat AVTA for 3 times and pick the set of vertices with highest quality where the quality is measured by sum of distances of each vertex to convex hull of other vertices. We set the number of output vertices $K = 50$ which is the same as the number of topics. i.e. each vertex corresponds to a topic. We found that initializing by simply assigning clusters using neighborhoods of highest degree vertices works effectively. As a final step, we use the post processing step from Bansal et al. (2014) to recover the topic-word matrix from the clustering.

Robustness We also generate perturbed version of the semi synthetic data. We generate a random matrix with i.i.d. entries uniformly distributed with different scales varying from 0.005–0.05. We test all the algorithms with the document-word matrix added with the noise matrix.

Results on semi synthetic data Figure 5d and e show the ℓ_1 reconstruction of all the four algorithms under both clean and noisy versions of the semi synthetic data set. For topic i, let A_i be the ground truth topic vector and $\widehat{A}_i$ be the topic vector recovered by the algorithm. Then the ℓ_1 error is defined as $\frac{1}{K}\sum_{i=1}^{K} \|A_i - \widehat{A}_i\|_1$. The plots show that AVTA+CatchWord is consistently better than both TSVD and Fast Anchor + Recoverl2 and produces significantly more accurate topic vectors. In order to further test the robustness of our approach, we plot in Fig. 5f the range of the ℓ_1 error obtained across multiple runs of the algorithms on the same data set. The range is defined to be the difference between the maximum and the minimum error recovered by the algorithm across different runs. We see that AVTA+CatchWord produces solutions that are much more stable to the effect of the noise as compared to other algorithms. Table 6 shows the running time of the experiments of 4 algorithms. As can be

Table 6 Running time of algorithms on semi synthetic data (secs)

Num of documents	5000	15, 000	30, 000	50, 000
Fast anchor + Recoverl2	5.49	6.00	10.30	13.60
AVTA + Recoverl2	7.82	7.68	12.84	16.40
TSVD	17.02	43.27	81.24	112.80
AVTA + Catch word	29.89	120.04	372.17	864.30

Table 7 Topic coherence on real data

	Fast anchor + RecoverL2	AVTA + RecoverL2	TSVD	AVTA + catch word
NIPS	-15.8 ± 2.24	-16.04 ± 2.09	-16.86 ± 1.66	-18.65 ± 1.78
NYTimes	-32.15 ± 2.7	-32.13 ± 2.43	-29.39 ± 1.43	-30.13 ± 1.98

Table 8 Running time on real data experiments (s)

	Fast anchor + RecoverL2	AVTA + RecoverL2	TSVD	AVTA + catch word
NIPS	3.22	4.41	56.58	22.78
NYTimes	26.05	27.79	237.6	101.07

seen, when using AVTA to learn topic models via the anchor words approach, our algorithm has comparable run time to Fast Anchor + Recoverl2. In CatchWord based learning, computing vertices is expensive compared to K-SVD step of TSVD thus AVTA has longer running time.

Results on real data Table 7 shows the topic coherence obtained by the algorithms. One can see that in both the approaches, either via anchor words or the clustering approach, AVTA based algorithms perform comparably to state of the art methods.[4] The running time is presented in Table 8. The AVTA+CathchWord has less running time in the real data experiments. Per our observation, the convex hull of word-document vectors in real data set has more vertices than K, the number of topics. The AVTA catches K vertices efficiently due to its small number of iterations on line search for γ. In semi-synthetic data set, the number of 'robust' vertices is approximately the same as number of topics K thus AVTA needs to find almost all vertices. To catch enough vertices, AVTA needs several iterations decreasing γ which is computationally expensive.

9.4 Non-negative matrix factorization

AVTA for NMF For our experiments on NMF we use the Swimmer data set (Donoho and Stodden 2003) that consists of 256 swimmer figures with each a 32×32 binary pixel images. One can interpret each image as a document and pixels as a word in the document (Ding et al. 2013). All swimmers consist of 4 limbs with each limb having 4 different possible poses. One can then consider the different poses of limbs as the true underlying topics (Donoho and Stodden 2003). We compare the algorithm proposed in Arora et al. (2012a) with AVTA+Recoverl2

[4] The topic coherence results for TSVD do not match the ones presented in Bansal et al. (2014) since in their experiments, the authors look at top 10 most frequent words in each topic. In our experiments we compute coherence for the top 5 most frequent words in each topic.

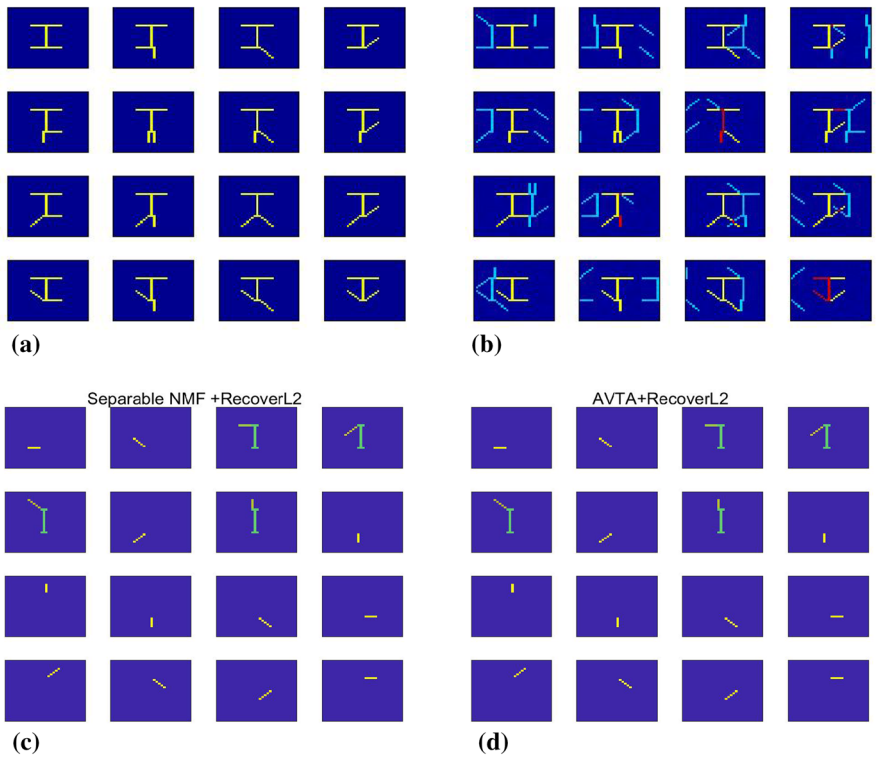

Fig. 6 **a** An example of swimmer images **b** An example of spurious actions in swimmer images **c** Output of NMF + RecoverL2 **d** Output of AVTA + RecoverL2

on the swimmer data set. We construct a noisy version by adding spurious poses to original swimmer data set. Let $\Omega(A)$ be a function that outputs a randomly chosen 32×8 block of an image. We generate a 'spurious pose' of size 32×8 by $\Omega(M_i)$ where M_i is a randomly chosen swimmer image. Then we take another randomly chosen image M_j and compute the corrupted image as $M'_j = M_j + c \cdot \Omega(M_i)$ where we simply set $c = 0.1$. An illustration of the noise data set is shown in Fig. 6b. Since the true underlying topics are known, we will plot the output of the algorithms and compare it with the underlying truth.

Results on NMF We compare the performance of AVTA on these data sets with the performance of the Separable NMF algorithm proposed in Arora et al. (2012a). Figure 6c and d show the output of the Separable NMF algorithm and that of our algorithm respectively on the noisy data set. Our approach produces competitive results as compared to the Separable NMF algorithm.

10 Conclusion

In this work we have presented a fast and robust algorithm for computing the vertices of the convex hull of a set of points. Our algorithm efficiently computes the vertices of convex hulls in high dimensions and even in the special case of the simplex is competitive with the

state of the art approaches in terms of running time (Arora et al. 2013). Furthermore, our algorithm leads to an improved algorithm for topic modeling that is more robust and produces better approximations to the topic-word matrix. It will be interesting to provide theoretical claims supporting this observation in the context of specific applications. Furthermore, we believe that our algorithm will have more applications in machine learning problems beyond the ones investigated here as well as applications in computational geometry and in linear programming.

Acknowledgements We wish to thank an anonymous reviewer for very constructive comments and suggestions.

Appendix

Efficient implementation of Triangle Algorithm

The worst-case complexity of each iteration in the Triangle Algorithm is $O(mn)$. Assuming that all the inner products $v_i^T v_j$ are computed the iteration complexity of Triangle Algorithm can be shown to reduce to $O(n)$. The cost of pre-computing the inner products is $O(mn^2)$. The algorithm can be made more efficient. To do so it suffices to compute the inner products $v_i^T v_j$ progressively rather than pre-computing them all. Ignoring this complexity, the iteration complexity of the Triangle Algorithm reduces to $O(N)$ where $N \leq n$ is the number of points of S considered in the Triangle Algorithm. The following shows how to achieve this.

Proposition 8 *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of $\mathbb{R}^m$. Consider testing if a given $p \in \mathbb{R}^m$ lies in $conv(\widehat{S})$. Suppose we have computed $\|p\|^2$, as well as $p^T\widehat{v}_i$, $i = 1, \ldots, N$. Suppose we have available $p' = \sum_{i=1}^N \alpha_i \widehat{v}_i \in conv(\widehat{S})$ satisfying $d(p', p) \leq \min\{d(p, \widehat{v}_i) : i = 1, \ldots, N\}$. Suppose $\|p'\|^2$ is also computed. Also, suppose $p'^T\widehat{v}_i$ is computed for each $i = 1, \ldots, N$. Then excluding the cost of computing the entries of the $N \times N$ matrix $\widehat{M} = (\widehat{v}_i^T\widehat{v}_j)$, each iteration of the Triangle Algorithm can be implemented in $O(N)$ operations. More precisely,*

(i) *Computation of a p-pivot $\widehat{v}_j$ at p', if one exists, takes $O(N)$ operations.*
(ii) *Given a pivot $\widehat{v}_j$, the computation of step size α takes $O(1)$ operations.*
(iii) *Computation of $Nearest(p; p'v) = p'' = (1-\alpha)p' + \alpha\widehat{v}_j = \sum_{i=1}^N \alpha_i'\widehat{v}_i$ takes $O(N)$ operations.*
(iv) *Computation of $\|p''\|^2$ takes $O(1)$ operations.*
(v) *Computation of $p''^T\widehat{v}_i$, $i = 1, \ldots, N$ takes $O(N)$ operations.*

Proof The Triangle Algorithm needs to use the entries of the $N \times N$ matrix $\widehat{M} = (\widehat{v}_i^T\widehat{v}_j)$. However, not all entries may be needed, nor do all entries of $\widehat{M}$ need to be computed in advance. Putting aside the complexity of computing $\widehat{M}$, in the following we justify the claimed complexities.

(i) From (10) and the given assumptions, to check if a particular $\widehat{v}_i$ is a pivot takes $O(1)$ operations. Thus to check if there exists a pivot takes $O(N)$ time.
(ii) From (12) and the assumptions, it suffices to compute $p^T p'$ and $p'^T\widehat{v}_j$ which takes $O(1)$ operations. (Note that p' is convex combination of $\widehat{v}_i$ and $p^T\widehat{v}_i$, $\widehat{v}_i^T\widehat{v}_j$ are computed.)
(iii) From equations (14) the computation of p'' and its representation as $p'' = \sum_{i=1}^N \alpha_i'\widehat{v}_i$ takes $O(N)$ operations.

(iv) Since $p'' = (1-\alpha)p' + \alpha\widehat{v}_j$, we have

$$\|p''\|^2 = p''^T p'' = (1-\alpha)^2\|p'\|^2 + 2\alpha(1-\alpha)p'^T\widehat{v}_j + \alpha^2\|\widehat{v}_j\|^2. \tag{94}$$

It follows that computing $\|p''\|^2$ takes $O(1)$ operations.

(v) Using that $p'' = (1-\alpha)p' + \alpha\widehat{v}_j$, the computation of $p''^T\widehat{v}_i$ takes $O(1)$ computations. Hence to compute all inner products $p''^T\widehat{v}_i$, $i = 1, \ldots, N$ takes $O(N)$ computations.

□

The following theorem combines Theorem 2 and Proposition 8 giving an improved complexity for the Triangle Algorithm.

Theorem 14 *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of $S = \{v_1, \ldots, v_n\}$. Given $p \in \mathbb{R}^m$, consider testing if $p \in conv(\widehat{S})$. Suppose $\|p\|^2$ as well as $p^T\widehat{v}_i$, $i = 1, \ldots, N$ are computed. Given $\varepsilon \in (0, 1)$, assume the Triangle Algorithm starts with $p' = \text{argmin}\{d(\widehat{v}_i, p) : i = 1, \ldots, N\}$. Then the complexity of testing if there exists an ε-approximate solution is*

$$O\left(mN^2 + \frac{N}{\varepsilon^2}\right). \tag{95}$$

In particular, suppose in testing if $p \in conv(S)$, $S = \{v_1, \ldots, v_n\}$, the Triangle Algorithm computes an ε-approximate solution p_ε by examining only the elements of a subset $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ of S. Then the number of operations to determine if there exists an ε-approximate solution $p_\varepsilon \in conv(S)$, is as stated in (95). □

References

Anandkumar, A., Foster, D. P., Hsu, D. J., Kakade, S. M. & Liu, Y.-K. (2012). A spectral algorithm for latent dirichlet allocation. In *Advances in neural information processing systems* (pp. 917–925).

Arora, S., Ge, R., Halpern, Y., Mimno, D., Moitra, A., Sontag, D., et al. (2013). A practical algorithm for topic modeling with provable guarantees. In *International conference on machine learning* (pp. 280–288).

Arora, S., Ge, R., Kannan, R., & Moitra, A. (2012a). Computing a nonnegative matrix factorization—provably. In *Proceedings of the forty-fourth annual ACM symposium on theory of computing* (pp. 145–162). ACM.

Arora, S., Ge, R., & Moitra, A. (2012b). Learning topic models—going beyond SVD. In *2012 IEEE 53rd annual symposium on foundations of computer science (FOCS)*, (pp. 1–10). IEEE.

Bansal, T., Bhattacharyya, C., & Kannan, R. (2014) A provable SVD-based algorithm for learning topics in dominant admixture corpus. In *Advances in neural information processing systems* (pp. 1997–2005).

Barber, C. B., Dobkin, D. P., & Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, *22*(4), 469–483.

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77–84.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Blum, A., Har-Peled, S., & Raichel, B. (2016). Sparse approximation via generating point sets. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on discrete algorithms* (pp. 548–557). Society for Industrial and Applied Mathematics.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121–167.

Chan, T. M. (1996a). Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, *16*(4), 361–368.

Chan, T. M. (1996b). Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete & Computational Geometry*, *16*(4), 369–387.

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, *10*(1), 377–409.

Chvatal, V. (1983). *Linear programming*. New York: Macmillan.

Clarkson, K. L. (1994). More output-sensitive geometric algorithms. In *1994 Proceedings of the 35th annual symposium on foundations of computer science* (pp. 695–702). IEEE.
Clarkson, K. L. (2010). Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, *6*(4), 63.
Ding, W., Rohban, M. H., Ishwar, P., & Saligrama, V. (2013). Topic discovery through data dependent and random projections. *ICML*, *3*, 1202–1210.
Donoho, D., & Stodden, V. (2003). When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in neural information processing systems*.
Frank, M., & Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics (NRL)*, *3*(1–2), 95–110.
Gärtner, B., & Jaggi, M. (2009). Coresets for polytope distance. In *Proceedings of the twenty-fifth annual symposium on computational geometry* (pp. 33–42). ACM.
Gilbert, E. G. (1966). An iterative procedure for computing the minimum of a quadratic form on a convex set. *SIAM Journal on Control*, *4*(1), 61–80.
Jaggi, M. (2013). Revisiting Frank–Wolfe: projection-free sparse convex optimization.
Jin, Y., & Kalantari, B. (2006). A procedure of chvátal for testing feasibility in linear programming and matrix scaling. *Linear Algebra and its Applications*, *416*(2–3), 795–798.
Johnson, W. B., & Lindenstrauss, J. (1984). Extensions of lipschitz mappings into a hilbert space. *Contemporary Mathematics*, *26*(189–206), 1.
Kalantari, B. (2015). A characterization theorem and an algorithm for a convex hull problem. *Annals of Operations Research*, *226*(1), 301–349.
Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on theory of computing* (pp. 302–311). ACM.
Khachiyan, L. G. (1980). Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, *20*(1), 53–72.
Lee, D. D. & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems* (pp. 556–562).
Stevens, K., Kegelmeyer, P., Andrzejewski, D., & Buttler, D. (2012). Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 952–961). Association for Computational Linguistics.
Toth, C. D., O'Rourke, J., & Goodman, J. E. (2004). *Handbook of discrete and computational geometry*. Boca Raton: CRC Press.
Vu, K., Poirion, P.-L., & Liberti, L. (2017). Random projections for linear programming. arXiv preprint arXiv:1706.02768.
Yao, L., Mimno, D., & McCallum, A. (2009). Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 937–946). ACM.
Zhang, T. (2003). Sequential greedy approximation for certain convex optimization problems. *IEEE Transactions on Information Theory*, *49*(3), 682–691.

Affiliations

Pranjal Awasthi[1] · Bahman Kalantari[1] · Yikai Zhang[1]

Pranjal Awasthi
pranjal.awasthi@cs.rutgers.edu

Yikai Zhang
yz422@cs.rutgers.edu

[1] Rutgers University, Piscataway, USA