How Slow is the *k*-Means Method?

David Arthur *
Stanford University
Stanford, CA
darthur@cs.stanford.edu

Sergei Vassilvitskii†
Stanford University
Stanford, CA
sergei@cs.stanford.edu

ABSTRACT

The `k-means` method is an old but popular clustering algorithm known for its observed speed and its simplicity. Until recently, however, no meaningful theoretical bounds were known on its running time. In this paper, we demonstrate that the worst-case running time of `k-means` is *superpolynomial* by improving the best known lower bound from $\Omega(n)$ iterations to $2^{\Omega(\sqrt{n})}$.

Categories and Subject Descriptors:
F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

General Terms:
Algorithms, Theory.

Keywords:
K-means, Local Search, Lower Bounds.

1. INTRODUCTION

The `k-means` method is a well known geometric clustering algorithm based on work by Lloyd in 1982 [12]. Given a set of n data points, the algorithm uses a local search approach to partition the points into k clusters. A set of k initial cluster centers is chosen arbitrarily. Each point is then assigned to the center closest to it, and the centers are recomputed as centers of mass of their assigned points. This is repeated until the process stabilizes. It can be shown that no partition occurs twice during the course of the algorithm, and so the algorithm is guaranteed to terminate.

The `k-means` method is still very popular today, and it has been applied in a wide variety of areas ranging from computational biology to computer graphics (see [1, 6, 8] for some recent applications). The main attraction of the algorithm lies in its simplicity and its *observed* speed.

Indeed, the running time of `k-means` is well studied experimentally (see, for example, [7]). In their text on pattern classification, Duda et al. remark that, "In practice the number of iterations is generally much less than the number of points" [5]. However, few meaningful theoretical bounds on the worst-case running time of `k-means` are known.

1.1 Related Work

There is a trivial upper bound of $O(k^n)$ iterations since no partition of points into clusters is ever repeated during the course of the algorithm. In d-dimensional space, this bound was slightly improved by Inaba et al. [9] to $O(n^{kd})$ by counting the number of distinct Voronoi partitions on n points. More recently, Dasgupta [4] presented some tighter results for a few special cases. He demonstrated a worst-case lower bound of $\Omega(n)$ iterations, and an upper bound of $O(n)$ for $k < 5$ and $d = 1$.

This work was extended by Har-Peled and Sadri [7] in 2005. Again restricting to $d = 1$, the authors show an upper bound of $O(n\Delta^2)$ where Δ is the spread of the point set (defined as the ratio between the largest pairwise distance and the smallest pairwise distance). They are unable to bound the running time of `k-means` in general, but they suggest a few modifications that are easier to analyze. For example, if one reclassifies exactly one point per iteration, then `k-means` is guaranteed to converge after $O(kn^2\Delta^2)$ iterations in any dimension.

1.2 Our Results

Our main result is a lower bound construction for which the running time of `k-means` is *superpolynomial*. In particular, we present a set of n data points and a set of adversarially chosen cluster centers for which the algorithm requires $2^{\Omega(\sqrt{n})}$ iterations. We then expand this to show that even if the initial cluster centers are chosen uniformly at random from the data points, the running time is still superpolynomial with high probability. We also show our construction can be modified to have constant spread, thereby disproving a recent conjecture of Har-Peled and Sadri [7].

Explaining the running times observed in practice remains an open problem. As a first step, we show that if the data points are selected from independent normal distributions in $\Omega(n/\log n)$ dimensions, then `k-means` will terminate in a polynomial number of steps with high probability. We also briefly discuss several other ways in which one might hope to circumvent the worst-case lower bound.

*Supported in part by an NDSEG Fellowship, NSF Grant ITR-0331640, and grants from Media-X and SNRC.

†Supported in part by NSF Grant ITR-0331640, and grants from Media-X and SNRC.

SCG'06, June 5–7, 2006, Sedona, Arizona, USA.

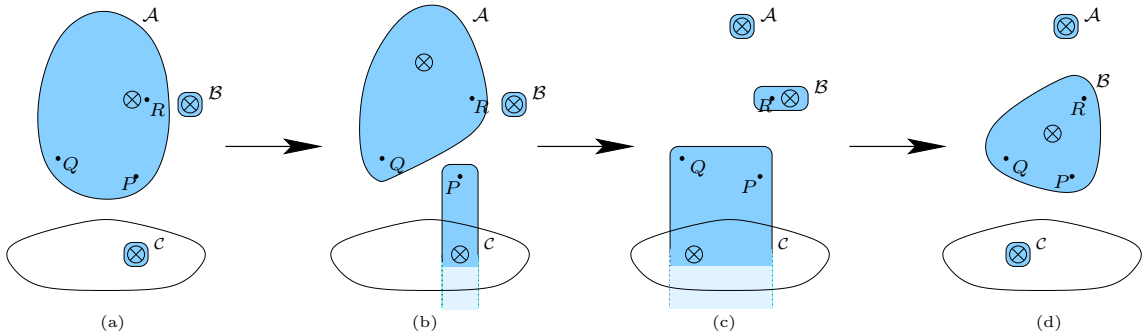

Figure 1: An idealized "reset widget" that can be used to reset the center of some cluster $\mathcal{C}$ after `k-means` has finished executing: (a) The configuration right before the signaling begins. (b) P switches to cluster $\mathcal{C}$, and the center of $\mathcal{A}$ moves away from Q and R. (c) Q switches to cluster $\mathcal{C}$, thereby resetting the center of $\mathcal{C}$. In addition, R switches to cluster $\mathcal{B}$, and the center of $\mathcal{B}$ moves towards P and Q. (d) P and Q switch to cluster $\mathcal{B}$. Now $\mathcal{C}$ is completely reset.

2. PRELIMINARIES

The `k-means` algorithm [12] is a method for partitioning data points into clusters. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of points in $\mathbb{R}^d$. After being seeded with a set of k centers $c_1, c_2, \ldots, c_k$ in $\mathbb{R}^d$, the algorithm partitions these points into clusters as follows.

1. For each $i \in \{1, \ldots, k\}$, set the cluster $\mathcal{C}_i$ to be the set of points in X that are closer to c_i than they are to c_j for all $j \neq i$.
2. For each $i \in \{1, \ldots, k\}$, set c_i to be the center of mass of all points in $\mathcal{C}_i$: $c_i = \frac{1}{|\mathcal{C}_i|} \sum_{x_j \in \mathcal{C}_i} x_j$.
3. Repeat steps 1 and 2 until c_i and $\mathcal{C}_i$ no longer change, at which point return the clusters $\mathcal{C}_i$.

If there are two centers equally close to a point in X, we break the tie arbitrarily. If a cluster has no data points at the end of step 2, we eliminate the cluster and continue as before. Our lower bound construction will not rely on either of these degeneracies.

During the analysis it will be useful to talk about a means configuration.

Definition 2.1. *A means configuration $M = (X, C)$ is a set of data points X and a set of cluster centers $C = \{c_i\}_{i=1,\ldots,k}$.*

Note that a means configuration M defines an intermediate point in the execution of the algorithm. Given a means configuration M, let $T(M)$ denote the number of iterations required by `k-means` to converge starting at M. We say that M is *non-degenerate* if, as the algorithm is run to completion, (a) no point is ever equidistant from the two closest cluster centers and (b) no cluster ever has 0 data points.

3. LOWER BOUNDS

In this section, we demonstrate a means configuration which requires $2^{\Omega(\sqrt{n})}$ iterations. As the construction is rather involved, we begin with some intuition, and then proceed with a formal proof. We have also implemented the construction in C++ [3].

At the end of the section, we consider a couple modifications to the main construction. First, we show that even if the starting centers are chosen uniformly at random from the data points, there exist examples where a superpolynomial number of iterations is still required with high probability. Finally, we show how to reduce the spread of any construction at the cost of increasing the dimensionality of the instance.

3.1 Intuition

The main idea behind the lower bound construction is that of a "reset widget". The role of the widget is to recognize when `k-means` has run to completion and to then reset it back into its initial state. We require the widget to not interfere with the original data points before or after the reset operation, thereby ensuring that the new `k-means` configuration takes twice as long to run to completion. Our lower bound is obtained by recursively adding reset widgets. By ensuring each widget has $O(k)$ new points and $O(1)$ new clusters, we get the bound of $2^{\Omega(\sqrt{n})}$ iterations.

We begin with an idealized description of a reset widget, illustrated in Figure 1. We then briefly mention a few issues that this idealized discussion omits.

3.1.1 A Reset Widget

Suppose we are given a means configuration in $\mathbb{R}^2$ and we are promised that the final center (c_x, c_y) of some cluster $\mathcal{C}$ never appears as a cluster center in any previous iteration. We call this a "signaling" means configuration. We can detect when `k-means` has run to completion by lifting the original configuration to $\mathbb{R}^3$, and adding a point $P = (c_x, c_y, D - \epsilon)$ in a new cluster $\mathcal{A}$ with center at $(c_x, c_y, 2D)$. If D is large and ϵ is small, then P will switch to $\mathcal{C}$ after `k-means` finishes executing on the original data set, but no earlier.

This creates a widget that triggers at the right time. The next step is to make the widget actually reset $\mathcal{C}$. We do this by augmenting $\mathcal{A}$ to also include a point $Q = (d_x, d_y, D(1 + \epsilon'))$ while maintaining the center of $\mathcal{A}$ at $(c_x, c_y, 2D)$. Switching P from $\mathcal{A}$ to $\mathcal{C}$ causes the center of $\mathcal{C}$ to move towards Q, and the center of $\mathcal{A}$ to move away from Q. As long as D is sufficiently large, Q will follow P into $\mathcal{C}$ on the next iteration, regardless of the values of d_x and d_y. In particular, this means we can choose d_x and d_y so as to reset the center of $\mathcal{C}$ to its initial position (at least in the x and y

coordinates). To avoid changing $\mathcal{C}$'s z coordinate, we make two symmetric reset widgets, one above the xy-plane, and one below. See Figure 1, parts (a) through (c).

Unfortunately, this method is not quite sufficient. We have reset the center of $\mathcal{C}$ by adding points to the cluster. As `k-means` progresses the second time through, these points will linger with $\mathcal{C}$ and provide a constant drag back to its original position. To actually make the reset configuration proceed as the original did, $\mathcal{C}$ must lose these new points immediately after the reset occurs.

To ensure this happens, we add a third point R near the center of $\mathcal{A}$, and a new cluster $\mathcal{B}$ near R. Now $\mathcal{B}$ will acquire R during the reset process, which moves it into position to recapture the points P and Q. The whole process is illustrated in Figure 1.

We have now fully reset $\mathcal{C}$. Applying this technique simultaneously to each cluster, we can hope to double the running time of `k-means`.

3.1.2 Pitfalls

A few additional considerations come into play when formalizing this intuition.

1. We can only add a reset widget to a signaling configuration. To recursively add reset widgets, we need to ensure that adding a reset widget maintains the signaling property.

2. We can only reset a cluster if that specific cluster signaled on the final iteration. Thus, we need to be able to take a signaling configuration and augment it so that each cluster simultaneously signals.

3. We cannot afford to double the number of clusters by adding a different reset widget for each cluster. Instead, we must have one widget reset all clusters at the same time. To accomplish this, it is convenient to have the reset widget cluster centered equally far from each signal, which requires placing certain cluster centers on a hypersphere.

3.2 The Formal Construction

We now formally present the reset widget. This requires a careful placement of points and cluster centers, but the intuition is exactly the one described above.

We first state our main results.

DEFINITION 3.1. *A means configuration is said to be* signaling *if at least one final cluster center is distinct from every cluster center arising in previous iterations.*

THEOREM 3.1. *Let M be a signaling, non-degenerate means configuration on n data points with k clusters. Then there exists a signaling, non-degenerate means configuration N on $n + O(k)$ data points with $k + O(1)$ clusters such that $T(N) \geq 2T(M)$.*

Starting with an arbitrary configuration, we can apply this construction t times to obtain a means configuration with $O(t^2)$ points and $O(t)$ clusters for which $T(M) \geq 2^t$. The superpolynomial complexity of `k-means` follows immediately.

COROLLARY 3.2. *The worst-case complexity of `k-means` on n data points is $2^{\Omega(\sqrt{n})}$.*

We prove Theorem 3.1 in two parts. First, we show that particular types of means configurations, called super-signaling configurations, can be slightly enlarged to create non-degenerate, signaling means configurations with twice the complexity. We then show how to slightly enlarge non-degenerate, signaling means configurations to obtain super-signaling configurations, thereby establishing the recursion.

DEFINITION 3.2. *A means configurations M is said to be* super-signaling *if it has the following properties.*

1. *The final positions of all cluster centers lie on a hypersphere.*
2. *The final positions of all cluster centers are distinct from all cluster centers arising in previous iterations.*
3. *There exists a means configuration M' with the same set of data points as M and with the same number of clusters as M. Furthermore, $T(M') = T(M)$ and at least one final cluster center in M' is distinct from any other cluster center arising in all iterations starting from M and M'.*

LEMMA 3.3. *Let M be a super-signaling, non-degenerate means configuration on n data points with k clusters. Then there exists a signaling, non-degenerate means configuration N on $n + O(k)$ data points with $k + O(1)$ clusters such that $T(N) \geq 2T(M)$.*

PROOF. We begin with a formal definition of our construction, and then trace the execution of `k-means` in Table 1 and Figures 3 - 7.

Let M' be given as in Definition 3.2. Label the clusters in M and M' with 1 through k, and let $x_{i,t}$ (respectively $y_{i,t}$) denote the center of cluster i in M (respectively M') after t iterations. Also let $\tilde{x}_i$ denote the final center of cluster i in M and let n_i denote the final number of data points in cluster i. Since M is super-signaling, we may assume without loss of generality that $\|\tilde{x}_i\|$ is independent of i (e.g. the center of the hypersphere passing through the x_i's lies at the origin). Finally, let $z_i = \frac{1}{2}((n_i + 4)y_{i,0} - (n_i + 2)\tilde{x}_i)$.

Let $V(M)$ denote the data points in M and let ℓ denote the diameter of $\{0, z_i, V(M)\}$. Let d, r and ϵ be such that $d \gg r \gg \ell \gg \epsilon > 0$ and let d' be such that $(d')^2 = d^2 + \|\tilde{x}_i\|^2 - \epsilon$. Finally, let $u_1, u_2, \ldots, u_k$ and $v_1, v_2, \ldots, v_k$ be vectors in $\mathbb{R}^2$ such that (a) $\|u_i\| = \frac{n_i+2}{2}$, (b) $v_i = \frac{u_i}{\|u_i\|}$, and (c) $v_i \neq v_j$ for all i, j.

Now consider the following points in $\mathrm{Span}(V(M)) \times \mathbb{R} \times \mathbb{R}^2 \times \mathbb{R}$,

$$\begin{aligned}
P_i &= (\tilde{x}_i, d', ru_i, 0) \text{ for } i \leq k,\\
P_i' &= (-\tilde{x}_i, d' + 2d, -ru_i, 0) \text{ for } i \leq k,\\
Q_i &= (z_i, d' + 0.001d, rv_i, 0) \text{ for } i \leq k,\\
Q_i' &= (-z_i, d' + 1.999d, -rv_i, 0) \text{ for } i \leq k,\\
A &= (0, d' + 0.99d, 0, 0),\\
A' &= (0, d' + 1.01d, 0, 0),\\
X &= (0, d' + 0.99d, 0, 0.2d),\\
X' &= (0, d' + 1.01d, 0, 0.2d).
\end{aligned}$$

For each such point Z, we define $\overline{Z}$ to be the reflection of P about the hyperplane $\mathrm{Span}(V(M)) \times \{0\} \times \mathbb{R}^2 \times \mathbb{R}$ — i.e. $\overline{P_i}$ has coordinates $(\tilde{x}_i, -d', ru_i, 0)$. Let $V(N)$ denote the set of all these points along with the natural embedding of $V(M)$

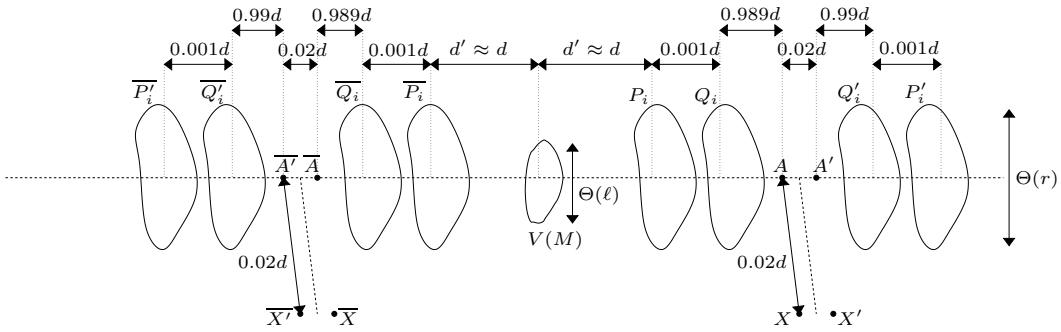

Figure 2: The data points constructed in Lemma 3.3 (Figure not to scale). Note $d \gg r \gg \ell$.

t	Clusters of N
0, …, T(M)	$\mathcal{C}_i = \mathcal{M}_{i,t}$ with center $= (x_{i,t}, 0, 0, 0)$ $\mathcal{G} = \{P_i, P_i', Q_i, Q_i', A, A'\}$ with center $= (0, d'+d, 0, 0)$ $\mathcal{H} = \{X\}$ with center $= (0, d'+0.99d, 0, 0.2d)$ $\mathcal{H}' = \{X'\}$ with center $= (0, d'+1.01d, 0, 0.2d)$
T(M)+1	$\mathcal{C}_i = \tilde{\mathcal{M}}_i \cup \{P_i, \overline{P_i}\}$ with center $= (\tilde{x}_i, 0, rv_i, 0)$ $\mathcal{G} = \{P_i', Q_i, Q_i', A, A'\}$ with center $(O(\ell), d'+\alpha d, O(rn), 0)$ with $1.25 \le \alpha \le 4/3$ $\mathcal{H} = \{X\}$ with center $= (0, d'+0.99d, 0, 0.2d)$ $\mathcal{H}' = \{X'\}$ with center $= (0, d'+1.01d, 0, 0.2d)$
T(M)+2	$\mathcal{C}_i = \tilde{\mathcal{M}}_i \cup \{P_i, Q_i, \overline{P_i}, \overline{Q_i}\}$ with center $= (y_{i,0}, 0, rv_i, 0)$ $\mathcal{G} = \{P_i', Q_i'\}$ with center $= (O(\ell), d'+1.9995d, O(rn), 0)$ $\mathcal{H} = \{A, X\}$ with center $= (0, d'+0.99d, 0, 0.1d)$ $\mathcal{H}' = \{A', X'\}$ with center $= (0, d'+1.01d, 0, 0.1d)$
T(M)+3	$\mathcal{C}_i = \mathcal{M}'_{i,1}$ with center $= (y_{i,1}, 0, 0, 0)$ $\mathcal{G} = \{P_i', Q_i'\}$ with center $= (O(\ell), d'+1.9995d, O(rn), 0)$ $\mathcal{H} = \{A, X, P_i, Q_i\}$ with center $= (O(\ell), d'+0.0005d+\frac{0.9895}{k+1}d, O(rn), \frac{d}{2k+2})$ $\mathcal{H}' = \{A', X'\}$ with center $= (0, d'+1.01d, 0, 0.1d)$
T(M)+4, …, 2T(M)+2	$\mathcal{C}_i = \mathcal{M}'_{i,t-T(M)-2}$ with center $= (y_{i,t-T(M)-2}, 0, 0, 0)$ $\mathcal{G} = \{P_i', Q_i'\}$ with center $= (O(\ell), d'+1.9995d, O(rn), 0)$ $\mathcal{H} = \{P_i, Q_i\}$ with center $= (O(\ell), d'+0.0005d, O(rn), 0)$ $\mathcal{H}' = \{A, A', X, X'\}$ with center $= (0, d'+d, 0, 0.1d)$

Table 1: The clusters of N after t iterations of k-means (see Lemma 3.3). $\mathcal{M}_{i,t}$ (respectively $\mathcal{M}'_{i,t}$) denotes the points in cluster of i of M (respectively M') after t iterations, and $\tilde{\mathcal{M}}_i$ denotes the final points in cluster i of M. All table entries describe clusters immediately after the centers are recomputed. Rather than going though every calculation, we discuss the key elements in the following figures.

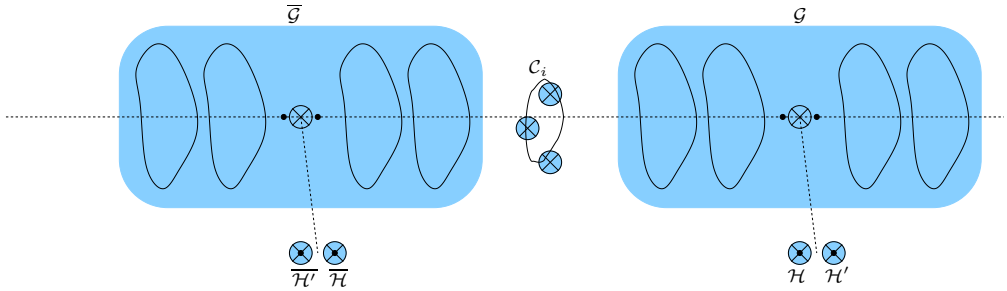

Figure 3: Clustering at $0 \le t \le T(M)$ (see Lemma 3.3). The clusters contained within $V(M)$ proceed independently of the other points. The remaining clusters are precarious but temporarily stable. For example, to see that P_i does not switch from cluster $\mathcal{G}$ to $\mathcal{C}_j$, note that the distance squared from P_i to the center of $\mathcal{C}_j$ minus the distance squared from P_i to the center of $\mathcal{G}$ is $(\|\tilde{x}_i - x_{j,t}\|^2 + (d')^2 + \|ru_i\|^2) - (\|\tilde{x}_i\|^2 + d^2 + \|ru_i\|^2) = \|x_i - x_{j,t}\|^2 - \epsilon > 0$. The last inequality follows from the fact that $\ell \gg \epsilon$ and that, since M is super-signaling, $\tilde{x}_i \neq x_{j,t}$.

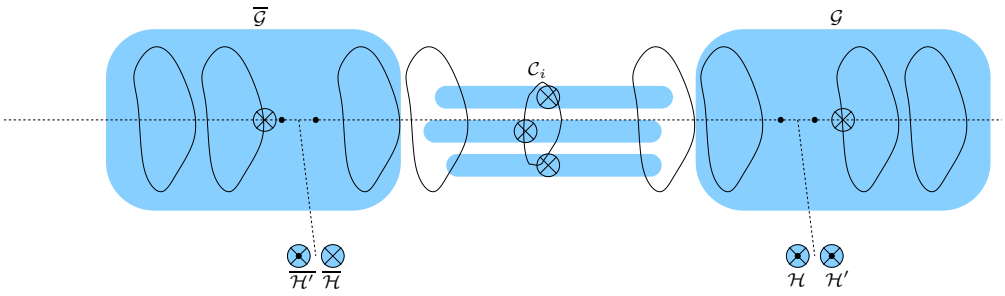

Figure 4: Clustering at $t = T(M) + 1$ (see Lemma 3.3). We now have $x_{i,t} = \tilde{x}_i$ for all i, and thus by the calculation in the previous step, each P_i switches to cluster $\mathcal{C}_i$. Clearly, this will result in a substantial shift of the center of $\mathcal{G}$ (and similarly of $\overline{\mathcal{G}}$). Furthermore, the u_i's have been chosen so that the center of $\mathcal{C}_i$ becomes $(\tilde{x}_i, 0, rv_i, 0)$.

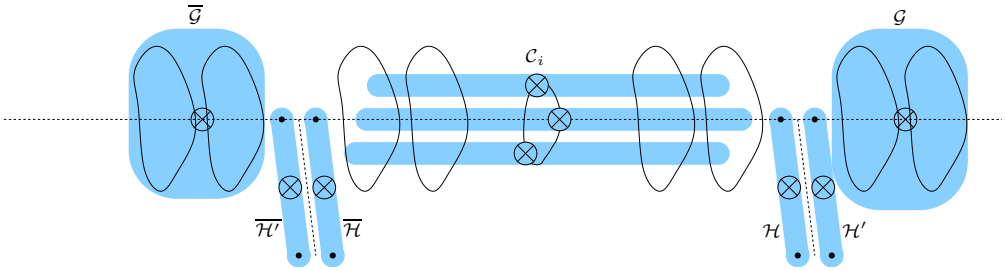

Figure 5: Clustering at $t = T(M) + 2$ (see Lemma 3.3). First consider $V(M)$. These points continue to be closer to the $\mathcal{C}_i$'s than to other clusters. Each $\mathcal{C}_i$ center has moved since the previous iteration, but they have all moved by a constant amount (namely $r\|v_i\| = r$) in a direction orthogonal to Span$(V(M))$. Therefore, the closest center to each point in $V(M)$ has not changed, and thus these points remain in their current clusters. On the other hand, since the center of $\mathcal{G}$ moved away, A, A', and Q_i all switch to different clusters. The first two clearly switch to $\mathcal{H}$ and $\mathcal{H}'$, but Q_i could reasonably switch to either $\mathcal{H}$ or any $\mathcal{C}_j$. The distance squared from Q_i to the center of $\mathcal{C}_j$ is $(1.001d)^2 + r^2\|v_i - v_j\|^2 + O(\ell^2)$, which is minimized when $i = j$. The distance squared from Q_i to the center of $\mathcal{H}$ is $(0.989d)^2 + (0.2d)^2 + O(r^2)$. Since $0.989^2 + 0.2^2 > 1.001^2$ and $d \gg r, \ell$, it follows that Q_i will in fact switch to $\mathcal{C}_i$.
Note that the analysis so far does not depend on the $V(M)$-coordinate of any Q_i, so we may choose those to make the $V(M)$-coordinate of each $\mathcal{C}_i$ equal to $y_{i,0}$ at the end of this step.

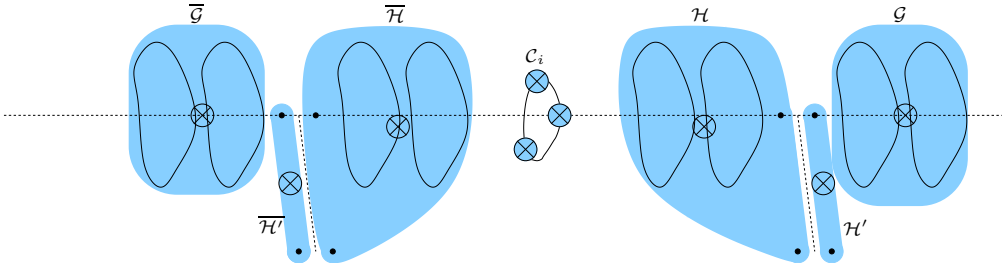

Figure 6: Clustering at $t = T(M) + 3$ (see Lemma 3.3). By acquiring A, cluster $\mathcal{H}$ has moved closer to the other points. In fact, the distance squared from P_i to the center of $\mathcal{H}$ is now $(0.99d)^2 + (0.1d)^2 + O(r^2) < d^2$. Thus, each P_i switches to $\mathcal{H}$, and a similar calculation shows each Q_i also switches to $\mathcal{H}$.
Now consider $V(M)$. As in the previous step, we may ignore the rv_i component of each $\mathcal{C}_i$ center. The $V(M)$ component of each $\mathcal{C}_i$ center is now $y_{i,0}$, which means the clustering proceeds according to M', and the points in $V(M)$ associated with $\mathcal{C}_i$ at the end of this step are $\mathcal{M}'_{i,1}$.

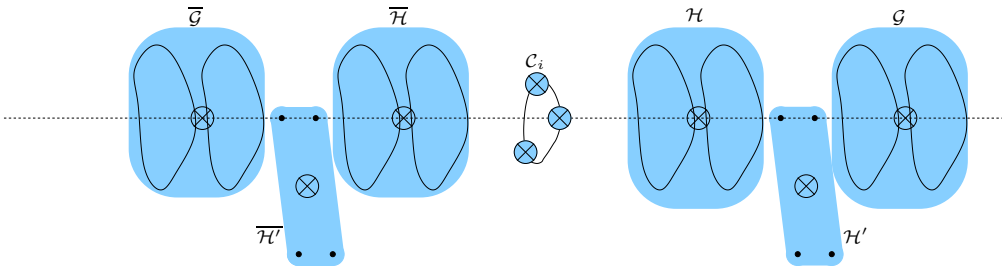

Figure 7: Clustering at $T(M)+4 \le t \le 2T(M)+2$ (see Lemma 3.3). The center of $\mathcal{H}$ moves because P_i and Q_i have been absorbed into $\mathcal{H}$. Also A and X switch to $\mathcal{H}'$. Beyond that, the configuration is now very stable, and the clustering on $V(M)$ will proceed normally according to M'.

in $\text{Span}(V(M)) \times \{0\} \times \{0,0\} \times \{0\}$. This setup is illustrated in Figure 2.

We also define clusters with initial centers in $\text{Span}(V(M)) \times \mathbb{R} \times \mathbb{R}^2 \times \mathbb{R}$ as follows.

$$\begin{aligned}
\mathcal{C}_i \text{ with center} &= (x_{i,0}, 0, 0, 0) \text{ for } i \le k,\\
\mathcal{G} \text{ with center} &= (0, d'+d, 0, 0),\\
\mathcal{H} \text{ with center} &= (0, d'+0.99d, 0, 0.2d),\\
\mathcal{H}' \text{ with center} &= (0, d'+1.01d, 0, 0.2d).
\end{aligned}$$

For each such cluster $\mathcal{C}$ other than the $\mathcal{C}_i$'s, we define $\overline{\mathcal{C}}$ to be a cluster whose initial center is obtained by reflecting the initial center of $\mathcal{C}$ about the hyperplane $\text{Span}(V(M)) \times \{0\} \times \mathbb{R}^2 \times \mathbb{R}$.

Let N denote the means configuration with all these cluster centers and with data points $V(N)$. We trace the evolution of `k-means` on N via Table 1 and Figures 3 - 7. Based upon this, we see that $T(N) \ge T(M) + T(M') = 2T(M)$, and that N is non-degenerate and signaling. Since N has $n + O(k)$ data points and $k + O(1)$ clusters, the result follows. □

This completes the first half of our construction, in which we transform a super-signaling configuration into a signaling configuration with twice the complexity. We now show how to transform a signaling configuration into a super-signaling configuration with equal complexity.

LEMMA 3.4. *Let N be a signaling, non-degenerate means configuration on n data points with k clusters. Then there exists a super-signaling, non-degenerate means configuration M on $n + O(k)$ data points with $k + O(1)$ clusters such that $T(M) \ge T(N)$.*

PROOF. Let $x_{i,t}$ denote the center of cluster i in N after t iterations and let $\tilde{x}_i$ denote the final center of cluster i in N. Since N is signaling, we may assume without loss of generality that $\tilde{x}_1$ is distinct from all other $x_{i,t}$. Let $V(N)$ denote the set of data points in N and let ℓ denote the diameter of $V(N)$. Let d and ϵ be such that $d \gg \ell \gg \epsilon$ and let d' be such that $(d')^2 = d^2 - \epsilon$. Also, let a, b and c be points in $V(N)$ such that $b = \frac{a+c}{2}$ and such that the distance from a to $V(N)$ is much larger than both ℓ and $\|c - a\|$.

Now, take $\alpha = \frac{1}{3k+9}$, and consider the following points in $\text{Span}(V(N)) \times \mathbb{R}$,

$$\begin{aligned}
P &= (\tilde{x}_1, d'),\\
X_i &= (\tilde{x}_i, d' + \alpha d) \text{ for } i \le k,\\
A, B, C &= (a,0), (b,0), (c,0),\\
A', B', C' &= (a, d'+\alpha d), (b, d'+\alpha d), (c, d'+\alpha d),\\
Q &= \left((k+4)\tilde{x}_1 - \sum \tilde{x}_i - 3b, d' + (k+14/3)d\right).
\end{aligned}$$

For each such point $Z \notin \{A, B, C\}$, we also define $\overline{Z}$ to be the reflection of Z about the hyperplane $\text{Span}(V(N)) \times \{0\}$. Let $V(M)$ denote the set of all these points as well as the natural embedding of $V(N)$ in $\text{Span}(V(N)) \times \{0\}$. This is illustrated in Figure 8.

We also define clusters with centers in $\text{Span}(V(N)) \times \mathbb{R}$ as follows.

$$\begin{aligned}
\mathcal{C}_i \text{ with center} &= (x_{i,0}, 0) \text{ for } i \le k,\\
\mathcal{H} \text{ with center} &= ((a+b)/2, 0),\\
\mathcal{H}' \text{ with center} &= (c, 0),\\
\mathcal{J} \text{ with center} &= (x_1, d'+d),\\
\overline{\mathcal{J}} \text{ with center} &= (x_1, -d'-d).
\end{aligned}$$

Let M denote the means configuration with all these cluster centers and with data points $V(M)$. We trace the evolution of `k-means` on M via Table 2 and Figures 9 - 11. Based upon this, we see that $T(M) \ge T(N)$, that M is non-degenerate, and also that the final cluster *sets* of M are distinct from all cluster sets arising in previous configurations.

Also let M' denote the means configuration with data points $V(M)$ and with cluster centers as above except with $\mathcal{H}$ centered at $(a, 0)$ and $\mathcal{H}'$ centered at $((b+c)/2, 0)$. Then, the same calculation shows that $T(M') = T(M)$ and that the final cluster set for $\mathcal{H}$ is distinct from all other cluster sets arising in M or M'.

Finally, since M and M' are non-degenerate, there exists a $\delta > 0$ such that we may move each data point by up to δ without altering the k-means execution. Taking advantage of this, we can ensure that the centers of distinct cluster sets are distinct, and that the final cluster centers of M' lie on a hypersphere. This makes M super-signaling, and the result follows. □

Theorem 3.1 follows immediately from Lemma 3.3 and Lemma 3.4.

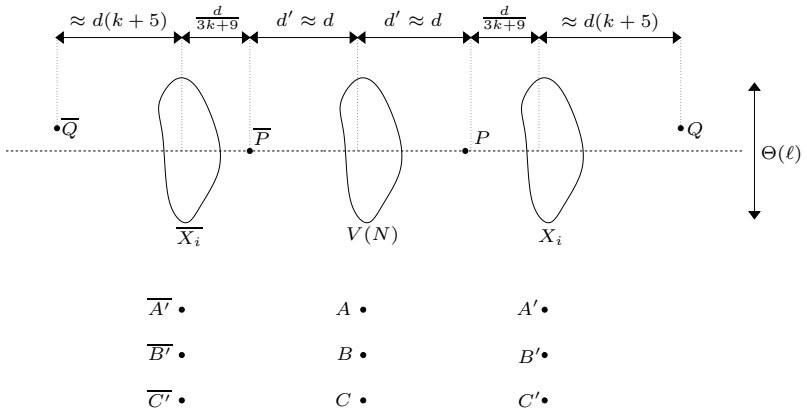

Figure 8: The data points constructed in Lemma 3.4. Note $d \gg \ell$.

t	Clusters of M
0, ..., T(N)	$\mathcal{C}_i = \mathcal{N}_{i,t}$ with center $= (x_{i,t}, 0)$ for $1 \le i \le k$ $\mathcal{H} = \{A, B\}$ with center $= (\frac{a+b}{2}, 0)$ $\mathcal{H}' = \{C\}$ with center $= (c, 0)$ $\mathcal{J} = \{P, X_i, A', B', C', Q\}$ with center $= (\tilde{x}_1, d' + d)$
T(N)+1	$\mathcal{C}_1 = \tilde{\mathcal{N}}_1 \cup \{P, \overline{P}\}$ with center $= (\tilde{x}_1, 0)$ $\mathcal{C}_i = \tilde{\mathcal{N}}_i$ with center $= (\tilde{x}_i, 0)$ for $2 \le i \le k$ $\mathcal{H} = \{A, B\}$ with center $= (\frac{a+b}{2}, 0)$ $\mathcal{H}' = \{C\}$ with center $= (c, 0)$ $\mathcal{J} = \{X_i, A', B', C', Q\}$ with center $= (\tilde{x}_1, d' + d + \frac{d}{k+4})$
T(N)+2	$\mathcal{C}_1 = \tilde{\mathcal{N}}_1 \cup \{P, X_1, \overline{P}, \overline{X_1}\}$ with center $= (\tilde{x}_1, 0)$ $\mathcal{C}_i = \tilde{N}_i \cup \{X_i, \overline{X_i}\}$ with center $= (\tilde{x}_i, 0)$ for $2 \le i \le k$ $\mathcal{H} = \{A, B, A', B', \overline{A'}, \overline{B'}\}$ with center $= (\frac{a+b}{2}, 0)$ $\mathcal{H}' = \{C, C', \overline{C'}\}$ with center $= (c, 0)$ $\mathcal{J} = \{Q\}$ with center $= ((k+4)\tilde{x}_1 - \sum \tilde{x}_i - 3b, d' + (k + 14/3)d)$

Table 2: The clusters of M after t iterations of k-means (see Lemma 3.4). $\mathcal{N}_{i,t}$ denotes the points in cluster of i of N after t iterations, and $\tilde{\mathcal{N}}_i$ denotes the final points in cluster i of N. All table entries describe clusters immediately after the centers are recomputed. Rather than going though every calculation, we discuss the key elements in the following pictures.

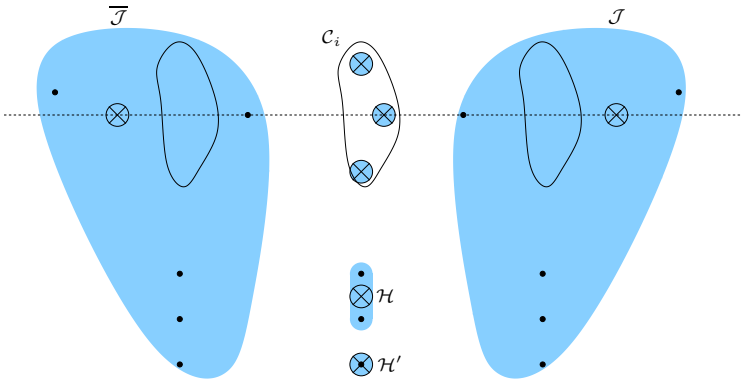

Figure 9: Clustering at $0 \le t \le T(N)$ (see Lemma 3.4). As with the first part of the construction for Lemma 3.3, the clusters contained within $V(N)$ proceed independently of the other points. The remaining clusters are precarious but temporarily stable. For example, to see that P does not switch from cluster $\mathcal{J}$ to $\mathcal{C}_i$, note that the distance squared from P to the center of $\mathcal{C}_i$ minus the distance squared from P to the center of J is $\|\tilde{x}_1 - x_{i,t}\|^2 + (d')^2 - d^2 = \|x_1 - x_{i,t}\|^2 - \epsilon > 0$. The last inequality follows from the fact that $\ell \gg \epsilon$ and that, since N is signaling, $\tilde{x}_1 \neq x_{i,t}$.

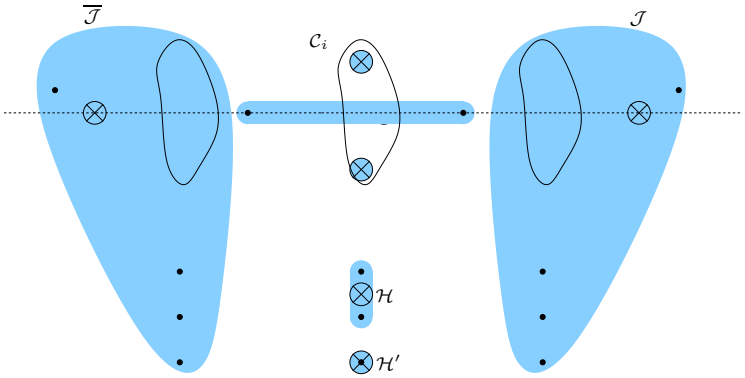

Figure 10: Clustering at $t = T(N) + 1$ (see Lemma 3.4). We now have $x_{1,t} = \tilde{x}_1$, and thus by the calculation in the previous step, P switches to cluster $\mathcal{C}_1$. Since $\overline{P}$ also switches to cluster $\mathcal{C}_1$, the center of $\mathcal{C}_1$ does not change. However, the centers of $\mathcal{J}$ and $\mathcal{J}'$ both move slightly further away from the other points.

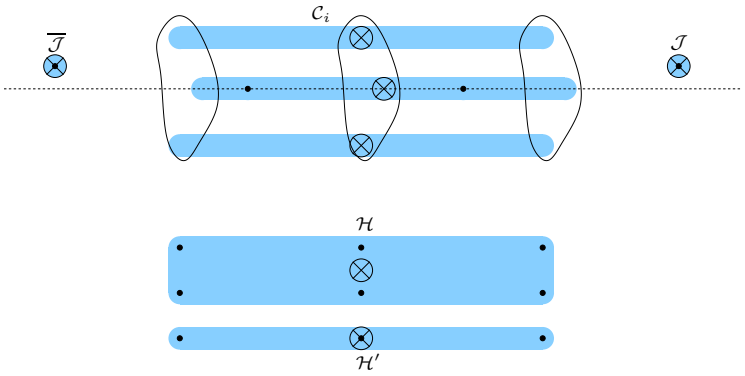

Figure 11: Clustering at $t = T(N) + 2$ (see Lemma 3.4). The points X_i, A', B', C' were all chosen to be only barely stable within $\mathcal{J}$. Thus, after the center of $\mathcal{J}$ moves, these points switch to the closest clusters in $V(N)$. For example, the distance from X_i to the center of $\mathcal{C}_i$ is approximately $d + \frac{d}{3k+9}$, and the distance from X_i to the center of $\mathcal{J}$ is approximately $d + \frac{d}{k+4} - \frac{d}{3k+9} > d + \frac{d}{3k+9}$ Again, only the centers of $\mathcal{J}$ and $\overline{\mathcal{J}}$ move as a result of this, and it is easy to check the new configuration is stable.

3.3 Probability Boosting

The construction used to prove Theorem 3.1 requires both a specific set of data points and a specific set of cluster centers. In practice, however, only the data points are specified and the initial cluster centers are chosen by the algorithm. Typically, the initial centers are chosen uniformly at random from the data points. Given this, one might ask if the superpolynomial lower bound can actually arise with non-vanishing probability.

In this section, we show how to modify our lower bound construction to apply with high probability even if the cluster centers are chosen randomly from the existing data points. It follows that `k-means` can still be very slow for certain sets of data points, even accounting for the random choice of cluster centers.

PROPOSITION 3.5. *Let M be a means configuration on n points. Then, there exists a set of $O(n^3 \log n)$ points such that if a means configuration N is constructed with these data points and with $4n \log n$ cluster centers chosen randomly from the set of data points, then $T(N) \geq T(M)$ with probability $1 - O(\frac{1}{n})$.*

PROOF. Let k be the number of clusters in M. For $i \leq k$ and $j \leq m$, let $u_{i,j}$ denote orthogonal unit vectors in $\mathbb{R}^{mk}$. Let $V(M)$ denote the set of data points in M and let ℓ denote the diameter of $V(M)$. Let d, r and ϵ be such that $d \gg r \gg \ell \gg \epsilon$. Also, let n_i denote the number of points in cluster i in M after one iteration. Replacing M with two identical overlapping copies if necessary, we may assume that $n_i > 1$. Finally, let $x_{i,t}$ denote the center of cluster i in M after t iterations.

Let m be a positive integer to be fixed later and consider the point set in $\text{Span}(V(M)) \times \mathbb{R}^{km} \times \mathbb{R}$ obtained by first embedding two copies of $V(M)$ at $\text{Span}(V(M)) \times \{0\} \times \{0\}$ and then adding the following points.

1. $P_{i,j} = (x_{i,0}, \sum_{(i',j') \neq (i,j)} r u_{i',j'}, d + j\epsilon)$ for $i \leq k, j \leq m$.
2. $Q_{i,\ell} = (\frac{n_i x_{i,1} - x_{i,0}}{n_i - 1}, \sum_{i' \neq i} \sum_{j'} r u_{i',j'}, d - \ell\epsilon)$ for $i \leq k$ and $\ell \leq n_i - 1$.
3. $O_j = (0, \sum_{i'} \sum_{j'} r u_{i',j'}, d + j\epsilon)$ for $j \leq m$.

Consider a means configuration N with these data points and with $4n \log n$ cluster centers chosen from these points at random. Let $\Gamma_0 = \{O_1, O_2, \ldots, O_m\}$ and for $i > 0$, let $\Gamma_i = \{P_{i,1}, P_{i,2} \ldots, P_{i,m}\}$. Suppose that N begins with all of its cluster centers in $\Gamma = \cup_i \Gamma_i$ and that each Γ_i has at least one cluster center. One can check that $T(N) \geq T(M)$ in this case.

Now, let $m = \frac{n^3 \log n}{k}$. Then, each cluster center will be in some Γ_i with probability $1 - O(\frac{1}{n^2 \log n})$. Since there are $4n \log n$ clusters, all clusters will be in Γ with probability $1 - O(\frac{1}{n})$. Furthermore, the probability that no cluster center is chosen in a fixed Γ_i is at most $(1 - \frac{1}{2k})^{4n \log n} \leq \frac{1}{n^2}$. Thus, each Γ_i has at least one cluster center with probability $1 - O(\frac{1}{n})$. The result now follows. □

3.4 Low spread

Recall the spread Δ of a point set is the ratio of the largest pairwise distance to the smallest pairwise distance. Har-Peled and Sadri [7] conjectured that `k-means` might run in time polynomial in n and Δ. In this section, however, we show that the spread can be reduced to $O(1)$ without decreasing the number of iterations required.

PROPOSITION 3.6. *Let M be a means configuration on n points. Then, there exists a means configuration N on $2n$ points such that N has $O(1)$ spread and such that $T(N) = T(M)$.*

PROOF. Let $V(M)$ denote the points in M, and choose an arbitrary set of vectors, $u_1, u_2, \ldots, u_n$. For each $v_i \in V(M)$, we replace v_i with $x_i = (v_i, u_i)$ and $y_i = (v_i, -u_i)$ in $\text{Span}(V(M)) \times \text{Span}(u_1, u_2, \ldots, u_n)$. Let N denote the means configuration with these data points and with centers $(c_j, 0)$ for each center c_j in M. It is easy to check that cluster $\mathcal{C}$ in N contains x_i and y_i after t iterations if and only if cluster $\mathcal{C}$ in M contains v_i after t iterations. It follows that $T(N) = T(M)$.

Taking u_i to be orthogonal and of length $d \gg 0$, we can make N have spread arbitrarily close to $\sqrt{2}$. □

More generally, there is a tradeoff between the extra dimensionality and the reduction of Δ. For example, by adding one extra dimension, and taking $u_i = di$, we can make the spread linear in n.

4. DISCUSSION

4.1 Smoothed Analysis

We have shown `k-means` can have a superpolynomial running time in the worst case. However, we know the algorithm runs efficiently in practice. It is natural to ask how this discrepancy can be formalized. One natural approach is that of smoothed analysis, which was used by Spielman and Teng [13] to explain the running time of the Simplex algorithm.

Towards that end, assume that the data points are chosen from independent, normal distributions with variance σ^2. Letting D denote the diameter of the resulting point set, we ask whether `k-means` is likely to run in time polynomial in n and $\frac{D}{\sigma}$.

4.1.1 High Dimension

This question appears to be difficult in general, but a positive result is relatively easy to prove in high dimensions. In this section, we sketch a proof of this fact.

PROPOSITION 4.1. *Given data points chosen from independent normal distributions with variance σ^2 and with dimension $d = \Omega(n/\log n)$, `k-means` will execute in polynomial time with high probability.*

We analyze the standard `k-means` potential function. For a means configuration $M = (X, C)$, let $\phi(M) = \sum_{i=1}^{n} \|x_i - c_i\|^2$, where $c_i \in C$ is the cluster center closest to x_i. Clearly, $0 \leq \phi \leq nD^2$, and one can also check that ϕ is non-increasing throughout an execution of `k-means`. Therefore, it suffices to show that the potential decreases by a non-trivial amount during each iteration.

On the one hand, it is known that if a cluster center moves by a distance δ during a `k-means` step and if the cluster has m points at the end of the step, then ϕ decreases by at least $\delta^2 m$ (see [7] and [11]). On the other hand, if our data points are random, no two possible centers can be too close. This can be formalized as follows.

DEFINITION 4.1. *We say a set of data points X is "ϵ-separated" if for any non-identical subsets S and T, the centers of mass $c(S)$ and $c(T)$ satisfy $\|c(S)-c(T)\| \geq \frac{\epsilon}{2\min(|S|,|T|)}$.*

LEMMA 4.2. *If X is a set of n data points chosen from independent normal distributions with variance σ^2, then X is ϵ-separated with probability at least $1 - 2^{2n}\left(\frac{\epsilon}{\sigma}\right)^d$.*

We omit the proof of the Lemma. Proposition 4.1 follows by choosing $\epsilon = \frac{\sigma}{n^{1/d}2^{2n/d}}$.

4.1.2 The General Case

Proposition 4.1 shows that `k-means` runs in polynomial time with high probability in smoothed high-dimensional settings. A similar result holds when $d = 1$ based on the spread analysis of [7] and the fact that a smoothed point set is likely to have polynomial spread.

A much more subtle analysis seems to be required for other values of d. We have recently proven an upper bound of $n^{O(k)} \cdot \text{poly}(n, \frac{D}{\sigma})$ [2], but it remains a major open problem to find a bound polynomial in n, k and $\frac{D}{\sigma}$ in the general case.

4.2 Variants

Smoothed analysis provides one very explicit way of circumventing the worst-case performance of `k-means`. Namely, given an arbitrary data set, we can perturb each point according to an independent normal distribution and then run `k-means`. Even our simple analysis in high dimensions can be harnessed here by first lifting to n-dimensional space, and then perturbing.

Two other methods also suggest themselves. First of all, `k-means` is often run in a relatively small number of dimensions, and regardless, one can always reduce d to $O(\log n)$ with small distortion [10]. Thus, it is natural to ask how `k-means` performs for small d. We conjecture that `k-means` is worst-case superpolynomial iff $d > 1$. Even when $d = 1$, no strongly polynomial upper bounds are known.

Finally, Har-Peled and Sadri [7] suggested a simple variant of `k-means` where only one data point is reassigned each iteration. This variant has running time polynomial in n and the spread Δ, which we know is not true for standard `k-means`. Given this qualitative improvement, a further study of this variant could prove fruitful.

5. REFERENCES

[1] Pankaj K. Agarwal and Nabil H. Mustafa. k-means projective clustering. In *PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 155–165, New York, NY, USA, 2004. ACM Press.

[2] David Arthur and Sergei Vassilvitskii. Improved smoothed analysis for the k-means method. Manuscript, 2006.

[3] David Arthur and Sergei Vassilvitskii. k-means lower bound implementation. `http://www.stanford.edu/~darthur/kMeansLbTest.zip`, 2006.

[4] Sanjoy Dasgupta. How fast is k -means? In *COLT Computational Learning Theory*, volume 2777, page 735, 2003.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[6] Frédéric Gibou and Ronald Fedkiw. A fast hybrid k-means level set algorithm for segmentation. In *4th Annual Hawaii International Conference on Statistics and Mathematics*, pages 281–291, 2005.

[7] Sariel Har-Peled and Bardia Sadri. How fast is the k-means method? *Algorithmica*, 41(3):185–202, 2005.

[8] R. Herwig, A.J. Poustka, C. Muller, C. Bull, H. Lehrach, and J O'Brien. Large-scale clustering of cdna-fingerprinting data. *Genome Research*, 9:1093–1105, 1999.

[9] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *SCG '94: Proceedings of the tenth annual symposium on Computational geometry*, pages 332–339, New York, NY, USA, 1994. ACM Press.

[10] W. Johnson and J. Lindenstrauss. Extensions of lipschitz maps into a hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[11] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *SCG '02: Proceedings of the eighteenth annual symposium on Computational geometry*, pages 10–18, New York, NY, USA, 2002. ACM Press.

[12] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–136, 1982.

[13] Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51(3):385–463, 2004.