k-means++: A Survey

Ryan Anselm, Edward Ri, Sam Wang

May 15, 2023

1 Introduction

In the standard formulation of the k-means clustering problem, one is given a number of clusters k and a set X of n data points in $\mathbb{R}^d$. The goal is to pick a set of cluster centers $\mathcal{C}$ that minimizes the cost function

$$\phi_X(\mathcal{C}) := \sum_{x \in X} \min_{c \in \mathcal{C}} ||x - c||^2 \tag{1}$$

such that $|\mathcal{C}| = k$. Denote ϕ_{OPT} to be the cost of an optimal clustering. Lloyd's algorithm [Llo82] (given in Algorithm 1) is a standard technique for obtaining locally optimal solutions to the k-means clustering problem. However, Lloyd's algorithm is known to be sensitive to initialization and can converge to arbitrarily bad clusterings (meaning the cost ratio $\frac{\phi_X(\mathcal{C})}{\phi_{\text{OPT}}}$ is potentially unbounded) when initial points are selected randomly (even under repeated random initializations).

Algorithm 1 Lloyd's algorithm for k-means clustering

```
Input: A set $X \subset \mathbb{R}^d$ of $n$ points, an integer $k > 0$ of the number of clusters.
Output: A set of clusters and cluster centers $(A_j, c_j)$ for $j \in [k]$.
1: Initialize centers $c_1, \ldots, c_k \in \mathbb{R}^d$ by picking $k$ points uniformly at random from $X$
2: while $c_1, \ldots, c_k$ have not yet converged do
3:     for each $j$: $A_j \leftarrow \{x \in X$ whose closest center is $c_j\}$
4:     for each $j$: $c_j \leftarrow \frac{1}{|A_j|} \sum_{x \in A_j} x$
5: return $\{(A_j, c_j) : j \in [k]\}$
```

The k-means++ initialization algorithm (given in Algorithm 2) was proposed by [AV06] as a replacement to step 1 of Lloyd's algorithm that picks the initial seeding according to a careful randomized procedure so as to obtain an expected $O(\log k)$-approximation of the optimal clustering, meaning $\frac{\phi_X(\mathcal{C})}{\phi_{OPT}} = O(\log k)$. k-means++ works by choosing the first cluster center uniformly at random and then picking all successive initial cluster centers with probabilities weighted proportionally to the distance squared from the closest cluster center already chosen so far.

k-means++ has inspired much follow-up work that attempts to better understand it and improve upon it, both theoretically and empirically. In this survey, we aim to give an overview of k-means++ as well as some of the most significant follow-up work related to it. Section 2 of this survey gives an overview of the properties of the standard k-means++ algorithm including proofs

Algorithm 2 k-means++ initialization

Input: A set $X \subset \mathbb{R}^d$ of n points, an integer $k > 0$ of the number of clusters.
Output: A set of initial cluster centers $\mathcal{C}$ such that $|\mathcal{C}| = k$.

```
1: $\mathcal{C} \leftarrow$ sample a point uniformly at random from $X$
2: while $|\mathcal{C}| \leq k$ do
3:     Sample $x \in X$ with probability $\frac{D^2(x,\mathcal{C})}{\phi_X(\mathcal{C})}$
4:     $\mathcal{C} \leftarrow \mathcal{C} \cup \{x\}$
5: return $\mathcal{C}$
```

of its theoretical guarantees and negative results related to it. Section 3 covers extensions of the k-means++ algorithm in a few different directions that improve upon some of the original's shortcomings.

2 Properties of the k-means++ algorithm

2.1 Performance guarantees of k-means++

While Lloyd's algorithm is simple to understand and requires very few iterations to converge in practice, one major pitfall is that not only is the final result is highly sensitive to initialization but the cost is unbounded from above. The motivation for the k-means++ algorithm is to resolve these issues, and the original paper proposes an upper bound on the expectation of the cost.

Theorem 1. *Let $\mathcal{C}_{OPT}$ be the optimal clustering and $\mathcal{C}$ be the cluster centers returned by the k-means++ algorithm. Then,*

$$\mathbb{E}[\phi_X(\mathcal{C})] \leq 8(\ln k + 2)\phi_X(\mathcal{C}_{OPT}). \tag{2}$$

Definition 1. *The cost induced on a subset $S \subseteq X$ by a cluster center c is $\phi_S(\{c\}) := \sum_{x \in S} ||x - c||^2$.*

Let $\mathcal{A}_{OPT} = \{A_1, \ldots, A_k\}$ be the set of clusters induced by the optimal centers $\mathcal{C}$. It can be "easily" shown that choosing a uniformly random point from cluster $A \in \mathcal{A}_{OPT}$ as a cluster center induces a cost over A which is on expectation 2 times the cost induced by the optimal cluster center μ_A over A, where $\mu_A := \frac{1}{|A|} \sum_{x \in A} x$. In essence, the proof of Theorem 1 boils down to showing that k-means++ is likely to choose a point from each of the optimal clusters, and applies the following lemma to all partitions to achieve the theorem result.

Lemma 1. *For any cluster A in $\mathcal{A}_{OPT}$, the cost induced on A by choosing a uniformly random point $a \in A$ as the cluster center will satisfy:*

$$\mathop{\mathbb{E}}_{a \in A}[\phi_A(\{a\})] = 2\phi_A(\{\mu_A\}). \tag{3}$$

Proof Sketch. Consider a set of points A. The optimal cluster center for the set of points is $\mu_A = \frac{1}{|A|}\sum_{x\in A} x$. Notice that

$$\begin{aligned}\mathop{\mathbb{E}}_{a\in A}[\phi_A(\{a\})] &= \frac{1}{|A|}\sum_{x\in A}\Big(\sum_{a\in A}\|a-x\|^2\Big)\\ &= \frac{1}{|A|}\sum_{x\in A}\Big(\sum_{a\in A}\|a-\mu_A\|^2 + |A|\cdot\|x-\mu_A\|^2\Big)\\ &= \frac{2}{|A|}\sum_{x\in A}\Big(\sum_{a\in A}\|a-\mu_A\|^2\Big)\\ &= 2\phi_A(\{\mu_A\}).\end{aligned}$$

The messy algebraic details of the second line are omitted but follow from the relationship between the distance of points to the mean and mutual distances between points. □

Notice that the above lemma implies that the first cluster center we choose will always abide by the given results, as we are guaranteed to be choosing a cluster center from a cluster in $\mathcal{A}_{OPT}$ without overlap.

Now, we show that as long as we choose centers from each cluster in $\mathcal{A}_{OPT}$, adding another point to $\mathcal{C}$ with the procedure used in k-means++ (known as D^2 weighting) will result in a similar bound for the total cost.

Lemma 2. *For an arbitrary cluster $A \in \mathcal{A}_{OPT}$ where a cluster center from A has not been sampled yet, adding a random point $x \in A$ to $\mathcal{C}$ chosen with weighting proportional to $D(x,\mathcal{C})^2$ will satisfy:*

$$\mathop{\mathbb{E}}_{x\sim D(\mathcal{C})^2}[\phi_A(\{x\})|x\in A] \le 8\cdot\phi_A(\{\mu_A\}) \tag{4}$$

where the distance to nearest cluster center squared weighted probability distribution is denoted as $D(\mathcal{C})^2$.

Proof. First, the probability that we choose some fixed point $x \in A$ as our new center given that we are choosing some point from A can be written as $\frac{D(x,\mathcal{C})^2}{\sum_{a\in A} D(a,\mathcal{C})^2}$ by definition of D^2 sampling. Notice that after choosing x as a new cluster center, the cost contributed by each point $a \in A$ will be equal to $\min(D(a,\mathcal{C})^2, \|a-x\|^2)$, as either a's new closest center will be x or will remain as an old cluster center in $\mathcal{C}$.

Thus, the contribution to the total expected cost by A can be written as:

$$\mathop{\mathbb{E}}_{x\sim D(\mathcal{C})^2}[\phi_A(\{x\}) \mid x\in A] = \sum_{x\in A}\frac{D(x,\mathcal{C})^2}{\sum_{a\in A} D(a,\mathcal{C})^2}\sum_{a\in A}\min(D(a,\mathcal{C})^2, \|a-x\|^2). \tag{5}$$

Notice that $D(x,\mathcal{C}) \le D(a,\mathcal{C}) + \|a-x\|$ for all a, x by triangle inequality and $D(x,\mathcal{C})^2 \le 2D(a,\mathcal{C})^2 + 2\|a-x\|^2$ because $(A+B)^2 \le 2(A^2+B^2)$. Applying these inequalities to the above equation, we get:

$$\mathop{\mathbb{E}}_{x\sim D(\mathcal{C})^2}[\phi_A(\{x\})|x\in A] \leq \frac{2}{|A|}\sum_{x\in A}\frac{\sum_{a\in A} D(a,\mathcal{C})^2}{\sum_{a\in A} D(a,\mathcal{C})^2}\sum_{a\in A}\min(D(a,\mathcal{C}),\|a-x\|)^2 + \sum_{a\in A}\frac{\sum_{a\in A}\|a-x\|^2}{\sum_{a\in A} D(a,\mathcal{C})^2}\sum_{a\in A}\min(D(a,\mathcal{C}),\|a-x\|)^2.$$

Now, notice that $\min(D(a,\mathcal{C}),\|a-x\|)^2 \leq D(a,\mathcal{C})^2$ and $\min(D(a,\mathcal{C}),\|a-x\|)^2 \leq \|a-x\|^2$ hold true, applying lemma 1 in the above equation and the above expressions gets us that

$$\begin{aligned}\mathop{\mathbb{E}}_{x\sim D(\mathcal{C})^2}[\phi_A(\{x\})] &\leq \frac{4}{|A|}\cdot\sum_{x\in A}\sum_{a\in A}\|a-x\|^2\\ &= 8\sum_{a\in A}\|a-\mu_A\|^2\\ &= 8\cdot\phi_A(\{\mu_A\}).\end{aligned}$$

□

As mentioned, note that while the above lemma shows a bound on the expected cost for sequentially choosing cluster centers, the underlying assumption is prohibitive in that it requires the D^2 weighting procedure to always sample from a new partition in $\mathcal{A}$. Here, we will show that choosing u random centers with D^2 weighting will result in an upper bound resembling the result of our original theorem.

Lemma 3. *Let $\mathcal{A}$ be some clustering with centers $\mathcal{C}$. Pick $u > 0$ to be the number of uncovered cells from $\mathcal{A}_{OPT}$, and X_u be the corresponding points from these cells. Let $\mathcal{C}'$ be $\mathcal{C}$ after adding $t \leq u$ random centers to $\mathcal{C}$ with D^2 weighting. Then,*

$$\mathbb{E}[\phi_X(\mathcal{C}')] \leq (1+H_t)\Big(\phi_{X\setminus X_u}(\mathcal{C}) + 8\phi_{OPT}(X_u)\Big) + \frac{u-t}{u}\cdot\phi_{X_u}(\mathcal{C}), \tag{6}$$

where $H_t = 1+\frac{1}{2}+...+\frac{1}{t}$ denotes the harmonic sum.

The lemma can be proved by induction, showing that if the result holds for $(t-1,u)$ and $(t-1,u-1)$, then the results holds for (t,u). The specific details of the proof are omitted for brevity.

The result of the main theorem can be exactly derivied by setting $t = u = k-1$ and using the fact that $H_t \leq 1+\ln k$.

Remark 1. There are a number of follow-up works that sharpen the results provided by [AV06]. [MRS20] improves the bound in the main theorem to $\mathbb{E}[\phi_X(\mathcal{C})] \leq 5(\ln k+2)\phi_X(\mathcal{C}_{OPT})$. [Wei16] generalizes the result by considering the family of D^l weighting, choosing βk clusters where $\beta \geq 1$, and metric spaces.

Remark 2. [AV06] also show a matching lower bound of $\Omega(\log k)$ for a specific arrangement of points and proves a lower bound on the cost optimality in expectation. However, while the original paper shows the existence of a construction where the bound is asymptotically tight, an open problem remained on the average case performance, i.e. whether k-means++ yields a constant factor approximation with constant probability.

2.2 A bad instance of k-means++

[BR13] attempts to follow up on this question by showing that k-means++ achieves an approximation ratio no better than $(\frac{2}{3} - \epsilon) \cdot \log k$ on a specific bad construction for k-means++.

Construction. Define $\Delta > 0$ as some appropriate number, dependent on k and some $\delta \in (0, 2/3)$. We consider placing the cluster centers first and putting the data points in a way such that k-means++ will provide a suboptimal answer.

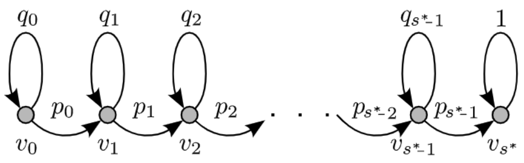
Concretely, place k centers $c_1, \dots, c_k$ such that they are $\sqrt{\Delta^2 - \frac{k-1}{k}}$ distance from each other in some high-dimensional space.[1] For each center c_i, we construct a $(k-1)$-simplex with the center as c_i and side length 1, and use the collection of vertices $x_1^{(i)}, \dots, x_k^{(i)}$ over all centers as the data points. Moreover, we assume we can place the simplices of all centers such that they are residing in orthogonal subspaces to each other.

By this construction, we can see that:

$$\begin{aligned} \|x_j^{(i)} - c_i\|^2 &= \frac{k-1}{2k}, \\ \|x_j^{(i)} - x_{j'}^{(i)}\|^2 &= 1, \\ \|x_j^{(i)} - x_{j'}^{(i')}\|^2 &= \Delta^2. \end{aligned}$$

Similar to the proof trajectory of [AV06], [BR13] consider the setup of sequentially choosing the next cluster center.

They model the procedure as a Markov chain, where each state denotes the coverage of each partition (cluster) in $\mathcal{C}_{\text{OPT}}$:



where $p_s = \frac{1}{1+\frac{s-1}{(k-s)\cdot\Delta^2}}$ and $q_s = 1 - p_s$. Notice that depending on the value of Δ (the distance between points from different cluster centers in the original construction), the difficulty of the problem changes. If Δ is too large, the probability that not all cells are covered becomes small as there is a high chance to jump to a different simplex using D^2 weighting if all simplices are mutually far away from each other. If Δ is too small, then the problem becomes intractable as recovering the original cluster centers becomes difficult.

3 Extensions of k-means++

The basic k-means++ algorithm encounters three limitations that are addressed by extensions to the original algorithm.

[1] In the original paper, constraints on the dimension are not mentioned; we will assume we have sufficient dimensions ($> k$) such that there is no issue in arranging points in the way described in the paper.

1. k-means++ is highly sensitive to outliers in the data. Recall that the probability of choosing any point to be a center is proportional to its distance from the nearest already chosen center squared, so outlier points are much more likely to be selected. In the extreme case, if we have 99% of points extremely close together with 1% of points far away, then most of the chosen centers will fall among the outliers.
2. k-means++ is not easily scalable for use on massive datasets because it is inherently sequential and cannot be run in a parallel setting. Specifically, k-means++ must make $O(k)$ sequential passes over the data because the weighted probability distribution for picking the ith center is only determined after the first $i-1$ centers have already been chosen.
3. The basic k-means++ algorithm only guarantees a $O(\log k)$ approximation factor in expectation. Ideally, we would prefer an algorithm that is a constant-factor approximation of the optimal result.

In this section, we discuss a few variants of the k-means++ procedure to overcome these shortcomings. In particular, robust k-means++ chooses centers using a combination of D^2 and random sampling to reduce the influence of outliers, k-means$\|$ over-samples centers to achieve results on par with k-means++ with a procedure that can be performed in a parallel setting, K-MC^2 speeds up the D^2 weighted sampling step by approximating it with a Markov Chain Monte Carlo (MCMC) sampling scheme, and local search iteratively samples new points and replaces existing points if the new one yields a lower cost function.

3.1 Robust k-means++

The k-means objective $\phi_X(\mathcal{C})$ is inherently sensitive to outliers. [DKP20] introduced the *robust* k-means objective $\rho_X(\mathcal{C},\beta)$[2] which discards a fraction β of outlier points:

$$\rho_X(\mathcal{C},\beta) = \min_{Y\subset X, |Y|=(1-\beta)|X|} \sum_{x\in X} \min_{c\in\mathcal{C}} ||x-c||^2. \tag{7}$$

Let $\mathcal{C}_{OPT} = \arg\min_{\mathcal{C}} \rho_X(\mathcal{C},\beta)$. Using a combination of D^2 and random sampling, the robust k-means++ algorithm returns a set S of $O(k/\delta)$ centers such that $\rho_X(S,\beta+\delta)$ is a constant factor approximation to $\rho_X(\mathcal{C}_{OPT},\beta)$. Whereas D^2 sampling is more likely to choose outlier points, uniform sampling is likely to miss small clusters, so a combination of approaches allows the best of both worlds.

The error parameter δ should be chosen such that each outlier point is likely to get assigned its own cluster. To output only k centers, we consider all $\binom{O(kn/z)}{k}$ ways of choosing k centers from the outputted points and return the optimal one, which runs in time linear in n and d but exponential in k.

Theorem 2. *Let $S \subset X$ be the subset of centers chosen by the robust k-means++ algorithm. Then, with constant probability, S contains k centers $\mathcal{C}_S$ such that $\rho_X(\mathcal{C}_S,\beta+\delta) \le 5\rho_X(\mathcal{C}_{OPT},\beta)$.*

Proof. Let $A_1,\ldots,A_k$ be the ideal clusters for the robust objective (so $\sum_{j=1}^{k}|A_j| = (1-\beta)n$) with

[2]This notation differs from the one used in the paper for the sake of clarity.

Algorithm 3 The robust k-means++ algorithm

Input: A set $X \subset \mathbb{R}^d$ of n points, an integer $k > 0$ of the number of clusters, an outlier parameter $\beta \in [0, 1]$, and an error parameter $\delta \in [0, 1]$.
Output: A set $S \subset X$ of size $O(k/\delta)$.

```
1: Initialize $S_0 \leftarrow \emptyset$.
2: for $i = 1, \ldots, t = O(k)$ do
3:     Sample $m = O(1/\delta)$ points $x_1, \ldots, x_m$ each with probability
           $\frac{\phi_{\{x\}}(S_{i-1})}{2\phi_X(S_{i-1})} + \frac{1}{2n}$
4:     $S_i \leftarrow S_{i-1} \cup \{x_1, \ldots, x_m\}$.
5: return $S_t$
```

centers $\mu_1, \ldots, \mu_k$. At the ith iteration of the loop, define

$$\begin{aligned} \mathrm{Good}_i &= \{A_j : \phi_{A_j}(S_{i-1}) \leq 5\phi_{A_j}(\{\mu_j\})\} \\ \mathrm{Bad}_i &= \{A_1, \ldots, A_k\} \setminus \mathrm{Good}_i. \end{aligned}$$

Good_i represents the set of optimal clusters that are approximated within a factor of 5 by S_{i-1}, and Bad_i is the set of clusters that remain to be approximated well. At each iteration i, there are two possibilities. If the number of points in bad clusters $\sum_{j:A_j \in \mathrm{Bad}_i} |A_j|$ is less than or equal to δn, then we get a 5-approximation after discarding $(\beta + \delta)n$ outlier points from the dataset X, and we are done (recall that $A_1 \cup \ldots \cup A_k$ already have a β fraction of points removed). Otherwise, the total number of bad clusters decreases with at least a constant probability, which is the statement of lemma 4:

Lemma 4. *Suppose* $\sum_{j:A_j \in Bad_i} |A_j| \geq \delta n$, *then after each iteration,* $\Pr[|Bad_{i+1}| < |Bad_i|] \geq \theta$ *for some* $\theta > 0$.

Proof. Let $B_j \subseteq A_j$ be the set of points in A_j within $\sqrt{2}$ times the root-mean-square radius of A_j:

$$r_j = \sqrt{\phi_{A_j}(\{\mu_j\})/|A_j|} \qquad B_j = \{x \in A_j : ||x - \mu_j|| \leq \sqrt{2} r_j\}.$$

First, note that x is a good cluster center if it is within $2r_j$ of μ_j:

$$\begin{aligned} \phi_{A_j}(S_i \cup \{x\}) &\leq \phi_{A_j}(\{x\}) \\ &= \phi_{A_j}(\{\mu_j\}) + |A_j| \, ||x - \mu_j||^2 \\ &\leq \phi_{A_j}(\{\mu_j\}) + |A_j| \, 4r_j^2 \\ &\leq 5\phi_{A_j}(\{\mu_j\}). \end{aligned}$$

Therefore, if we have a bad cluster A_j in the ith iteration and we choose some $x \in B_j$ to include in S_{i+1}, then $A_j \in \mathrm{Good}_{i+1}$. What remains is to lower bound the probability that the chosen point falls in B_j for some bad cluster A_j. Since at least δn points belong to bad clusters,

$$\Pr[x \text{ is in a bad cluster}] = \frac{\sum_{j:A_j \in \mathrm{Bad}_i} \phi_{A_j}(S_i)}{2\phi_X(S_i)} + \frac{\sum_{j:A_j \in \mathrm{Bad}_i} |A_j|}{2n} \geq \frac{\sum_{j:A_j \in \mathrm{Bad}_i} |A_j|}{2n} \geq \frac{\delta}{2}.$$

For a fixed bad cluster A_j,

$$\Pr[x \in B_j \mid x \in A_j] = \left(\frac{\phi_{B_j}(S_i)}{2\phi_X(S_i)} + \frac{|B_j|}{2n}\right) \Big/ \left(\frac{\phi_{A_j}(S_i)}{2\phi_X(S_i)} + \frac{|A_j|}{2n}\right) \geq \min\left\{\frac{\phi_{B_j}(S_i)}{\phi_{A_j}(S_i)}, \frac{|B_j|}{|A_j|}\right\}.$$

Observe that $|B_j|/|A_j|$ must be at least $1/2$, or else the contribution of $A_j \setminus B_j$ to the cost would be too large. Let p be the closest point in S_i to μ_j and $d = ||p - \mu_j||$. Then,

$$\begin{aligned}\phi_{B_j}(S_i) &= \sum_{x \in B_j} \min_{s \in S_i} ||x - s||^2 \\ &\geq \sum_{x \in B_j} (||p - \mu_j|| - ||\mu_j - x||)^2 \\ &\geq |B_j|(d - \sqrt{2}r_j)^2 \geq \frac{|A_j|}{2}(d - \sqrt{2}r_j)^2.\end{aligned}$$

On the other hand,

$$\phi_{A_j}(S_i) \leq \phi_{A_j}(\{p\}) = \phi_{A_j}(\{\mu_j\}) + |A_j|\,||p - \mu_j||^2 \leq |A_j|(r_j^2 + d^2) \leq |A_j|(r_j + d)^2.$$

Putting these two expressions together, we get that

$$\frac{\phi_{B_j}(S_i)}{\phi_{A_j}(S_i)} \geq \frac{(d - \sqrt{2}r_j)^2}{2(r_j + d)^2} \geq \frac{(2 - \sqrt{2})^2}{10} \geq \frac{1}{30}.$$

This result relies on the fact that $d > 2r_j$ since $A_j \in \text{Bad}_i$. Finally, the probability of choosing some x such that x belongs to B_j of some bad cluster A_j is at least $(\delta/2)(1/30)$ which is $\delta/60$, so by sampling $O(1/\delta)$ points in each iteration, $|\text{Bad}_i|$ decreases between iterations with at least a constant probability. □

Definition 2. *A sequence of random variables $J_0, J_1, \ldots, J_t$ is called a super-martingale if $\mathbb{E}[J_{i+1} \mid J_0, \ldots, J_i] \leq J_i$ for all $i \geq 1$.*

The Azuma-Hoeffding inequality states that if $J_0, J_1, \ldots, J_t$ is a super-martingale with $J_{i+1} - J_i \leq 1$, then $\Pr[J_t \geq J_0 + \epsilon] \leq e^{-\epsilon^2/2t}$. Define an indicator variable X_i such that

$$X_i = \begin{cases} 1 & \text{if } |\text{Bad}_{i+1}| = |\text{Bad}_i| \\ 0 & \text{otherwise} \end{cases},$$

and define

$$J_i = \sum_{j=1}^{i}(X_j - (1 - \theta)) = i(1 - \theta) + \sum_{j=1}^{i} X_j.$$

Clearly, $\mathbb{E}[J_{i+1} \mid J_0, \ldots, J_i] \leq J_{i-1}$ and $J_{i+1} - J_i \leq 1$. Letting $t = (k + \sqrt{k})/\theta$ and $\epsilon = \sqrt{k}$, and applying the Azuma-Hoeffding inequality, we get that

$$\Pr[\text{no bad clusters after } (k + \sqrt{k})/\theta \text{ iterations}] = \Pr\left[\sum_{i=1}^{(k+\sqrt{k})/\theta}(1 - X_i) \geq k\right] \geq 1 - e^{-\theta/4}.$$

Therefore, after $O(k)$ iterations of the loop, the robust k-means++ algorithm terminates with a set S containing k centers that approximate $\mathcal{C}_{OPT}$ with probability at least $1 - e^{-\theta/4}$, as desired. □

The k-means++ algorithm runs in $O(ndk)$ time and outputs $O(k/\delta)$ centers. To narrow down the number of centers to k, we require an additional $O((ndk)\cdot(k/\delta)^k)$ time to find the optimal subset.

Remark 3. With $O(\log 1/\eta)$ runs of the algorithm, we can boost the success probability to $1-\eta$.

Remark 4. The given algorithm applies equal weighting to D^2 and uniform sampling, but this not be the case. We can apply an $(\alpha, 1-\alpha)$ weighting to favor one type of sampling based on the nature of the dataset.

3.2 Scaling k-means++ for massive data

While scaling and parallelizing Lloyd's algorithm for k-means clustering to massive data is relatively straightforward, the k-means++ initialization step is inherently sequential in nature and cannot apparently be sped up by parallelism, making it more of a time bottleneck in practice. This section covers extensions to k-means++ that try to make it more practical for use on massive datasets. We examine k-means$\|$, which reduces the number of sequential passes required with the tradeoff of doing extra within each pass, and K-MC^2, which improves the time complexity of each sequential pass through a Markov Chain Monte Carlo-based approximate sampling scheme.

3.2.1 k-means$\|$

[BMV$^+$12] introduced k-means$\|$ (given in Algorithm 4), a parallelizable version of k-means++ that significantly reduces the number of sequential passes required in practice to obtain a good initialization. The key modification is that instead of sampling a single point in each pass for $O(k)$ passes, k-means$\|$ samples $O(k)$ points in each pass for approximately $O(\log n)$ passes. At the end it is left with $O(k\log n)$ points that are reclustered into k centers using a clustering algorithm such as k-means++ followed by Lloyd's algorithm, which then become the initial points for the main run of Lloyd's algorithm. Like k-means++, this approach has provable approximation guarantees, but additionally it is parallelizable and applicable in practice to massive datasets due to requiring significantly fewer sequential operations.

Algorithm 4 The k-means$\|$ initialization algorithm

Input: A set $X \subset \mathbb{R}^d$ of n points, an integer $k > 0$ of the number of clusters, an *oversampling factor* $l = \Theta(k)$ on the order of k.
Output: A set $\mathcal{C}$ of k cluster centers.

```
1: $\mathcal{C} \leftarrow$ sample a point uniformly at random from $X$
2: $\psi \leftarrow \phi_X(\mathcal{C})$, the cost of the initial cluster
3: for $O(\log\psi)$ times do
4:     $\mathcal{C}' \leftarrow$ sample each point $x \in X$ independently with probability $p_x = \frac{l\cdot D^2(x,\mathcal{C})}{\phi_X(\mathcal{C})}$
5:     $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$
6: For $x \in \mathcal{C}$, set $w_x$ to be the number of points in $X$ closer to $x$ than any other point in $\mathcal{C}$
7: $\mathcal{C} \leftarrow$ cluster centers of $\mathcal{C}$ using some clustering algorithm
8: return $\mathcal{C}$
```

Lemma 5. *The expected number of points chosen per iteration of steps 3-5 is l, and on expectation there will be $l\cdot\log\psi = O(k\log n)$ points in $\mathcal{C}$ at the start of step 6 of Algorithm 4.*

To see this, observe that $\sum_{x \in X} p_x = l$ and $\psi \leq n^2 \Delta^2$ where Δ is the maximum interpoint distance in X. Then it follows from this and the fact that $l = \Theta(k)$.

Theorem 3. *After $O(\log \psi)$ passes of steps 3-5 in algorithm 4, there will be a set of (on expectation) $O(k \log n)$ points that gives on expectation a constant factor approximation of the optimal k-means clustering cost.*

Proof Sketch. Consider a cluster A present in the optimum k-means solution. Denote $\phi_A(\mathcal{C})$ to be the cost over points in A induced by a set of cluster centers $\mathcal{C}$. Let $T = |A|$ and sort points in A in increasing distance from the centroid of A, denoted $a_1, \ldots, a_T$. Let q_t be the probability a_t is chosen out of points in A first and q_{T+1} be the probability no point is sampled from A during an iteration of k-means$\|$. Recall that in k-means$\|$, $p_t = \frac{l \cdot D^2(a_t, \mathcal{C})}{\phi_X(\mathcal{C})}$. Observe that $q_t = p_t \prod_{j=1}^{t-1}(1 - p_j)$ and $q_{T+1} = 1 - \sum_{t=1}^{T} q_t$. Define a_t as the first point in A sampled as a new center and define $s_t = \min\{\phi_A, \sum_{a \in A} ||a - a_t||^2\}$. We have for a new set of points selected $\mathcal{C}'$ that

$$\mathbb{E}[\phi_A(\mathcal{C} \cup \mathcal{C}')] \leq \sum_{t=1}^{T} q_t s_t + q_{T+1} \phi_A(\mathcal{C}) \tag{8}$$

Consider a simplifying mean-field case where all points in A are far from the current clustering, and that all $d(a_t, \mathcal{C})$ are equal, from which it follows that $p := p_1 = p_2 = \cdots = p_T$.[3] Under these conditions we have that $q_t = p(1-p)^{t-1}$, and that $\{q_t\}_{1 \leq t \leq T}$ is a monotone decreasing sequence. We now define $s'_t := \sum_{a \in A} ||a - a_t||^2$. By the ordering on a_t's, $\{s'_t\}_{1 \leq t \leq T}$ is an increasing sequence. Then, we observe that

$$\sum_{t=1}^{T} q_t s_t \leq \sum_{t=1}^{T} q_t s'_t \leq \frac{1}{T} \left(\sum_{t=1}^{T} q_t \cdot \sum_{t=1}^{T} s'_t \right)$$

using Chebyshev's sum inequality [HLP88]. Finally, note that $\frac{1}{T} \sum_{t=1}^{T} s'_t = 2\phi(A, \mu_A)$. Then, by substituting into equation 8 we get

$$\mathbb{E}[\phi_A(C \cup C')] \leq (1 - q_{T+1}) 2\phi(A, \mu_A) + q_{T+1} \phi_A(\mathcal{C}). \tag{9}$$

This result implies that in each iteration, for each optimal cluster A, a fraction of the current cost $\phi_A(\mathcal{C})$ is removed and replaced with a constant factor times $\phi(A, \mu_A)$. Repeating this process $O(\log \psi)$ many times will therefore lower the cost on each of the optimal clusters to a constant factor approximation of each optimal cluster, and together they yield a constant factor approximation of the total cost using $l \log \psi$ points on expectation. $\square$

Corollary 1. *If an α-approximation clustering algorithm is used to recluster the $O(k \log n)$ points in Step 7, then k-means$\|$ obtains a solution that is an $O(\alpha)$-approximation to k-means.*

Note that k-means++ is an α-approximation algorithm on expectation for $\alpha = 5(\ln k + 2)$. This corollary implies that if we use k-means++ to cluster the reduced set of $O(k \log n)$ points in $\mathcal{C}$ into k clusters, then k-means$\|$ will be an $O(\log k)$-approximation algorithm to the full k-means problem.

[3]For an argument that works for the general case when p_t's are not necessarily the same, see [BMV+12].

Remark 5. While the overall time complexity of k-means$\|$ is higher than k-means++, its capacity for being parallelized results in it being more useful in practice for massive datasets. The set produced at the end of the iterations contains only $l \log \psi = O(k \log n)$ points on expectation rather than $O(n)$, which makes it practically feasible for the last reclustering stage to be done sequentially using standard k-means++, even for large n.

Remark 6. Interestingly, k-means$\|$ also outperforms k-means++ empirically in the single-machine setting despite having weaker theoretical guarantees. [MRS20] gives an empirical demonstration to suggest that this occurs because k-means$\|$ samples $O(k \log n) \geq k$ centers in its first stage and then reclusters them back down to k later. They show that the performance of k-means$\|$ where only k points are selected in the first stage of k-means$\|$ is nearly identical to k-means++. They furthermore compare k-means$\|$ to Bi-Criteria k-means++ with Pruning, which samples $k + \Delta$ centers before reducing down to k and finds that it is also nearly identical in performance to k-means$\|$.

3.2.2 K-MC^2 - Sublinear approximate k-means++

[BLHK16a] addressed the issue of k-means++'s lack of scalability from another angle by proposing a seeding algorithm called K-MC^2 that approximates the D^2-sampling in k-means++ with an approach based on Markov Chain Monte Carlo (MCMC) sampling. Under assumptions that the data comes i.i.d. from a distribution that has (1) exponential tails and (2) is approximately uniform on a hypersphere, their algorithm retains the full original approximation guarantees while gaining a sublinear in n time complexity. A single pass of approximate D^2-sampling works by first sampling a point x_0 uniformly at random from X, then building a Markov chain of length m by iteratively sampling a uniformly random candidate y_j and accepting it with probability

$$\pi = \min\left(1, \frac{D^2(y_j, \mathcal{C})}{D^2(x_{j-1}, \mathcal{C})}\right),$$

with probability π setting x_j to y_j and with probability $1-\pi$ setting x_j to x_{j-1}. The point that gets sampled at the end will be x_m. The speedup in this process is found in only having to compute the distance squared from $\mathcal{C}$ for m points per pass, rather than for the entire dataset of n points. This approximate D^2-sampling scheme can directly substitute the true D^2-sampling in the k-means++ algorithm. It can be shown that when the probability measures of the MCMC-based approximate scheme and k-means++ are 'close' (under a metric known as *total variation distance*) we retain the same theoretical guarantees as k-means++ with high probability.

Theorem 4 (Informal)**.** *The probability distributions of the MCMC sampling scheme and true k-means++ will be 'close' to each other for a Markov chain of length only sublinear in n.*[4]

Corollary 2. *Using the MCMC-based approximate-D^2 sampling scheme, we can obtain $O(\log k)$-competitive solutions with total time complexity $O(k^3 d \log^2 n \log k)$ for initialization (compare this to $O(nkd)$ for vanilla k-means++).*

Remark 7. [BLHK16b] (by the same authors) improves upon this result by giving a modified algorithm that does not require the assumptions on the data that K-MC^2 requires in order to achieve the same approximation guarantees as k-means++. We mention this result for the sake of completeness but do not detail it further.

[4]For particulars of this theorem, see [BLHK16a].

3.3 k-means++ with local search

To achieve a constant-factor approximation to the k-means problem, the idea is to improve the k-means++ initialization by sampling additional points using the same D^2 weighting. If we can replace an existing center $q \in \mathcal{C}$ with the newly sampled point p and decrease the overall cost, then we swap them out. If Z is the number of rounds of local search, then the total running time of the initialization procedure is $O(dnkZ)$.

Algorithm 5 k-means++ initialization with local search

```
1: $\mathcal{C} \leftarrow$ a set of $k$ centers by $k$-means++ initialization
2: **for** $t = 1, \ldots, Z$ **do**
3:     Sample $p \in X$ with probability $\frac{D^2(x,\mathcal{C})}{\phi_X(\mathcal{C})}$
4:     $q = \arg\min_{c \in \mathcal{C}} \phi_X(\mathcal{C} \setminus \{c\} \cup \{p\})$
5:     **if** $\phi_X(\mathcal{C} \setminus \{c\} \cup \{p\}) < \phi_X(\mathcal{C})$ **then**
6:         $\mathcal{C} \leftarrow \mathcal{C} \setminus \{q\} \cup \{p\}$
7: **return** $\mathcal{C}$
```

[LS19] proved that $O(k \log\log k)$ rounds of local search are sufficient to achieve a constant-factor approximation. [CGPR20] improved upon the result to show that only $O(k)$ rounds are sufficient. Although both papers proved theoretical bounds with large hidden constants, only a small number ($\approx k$) of iterations are needed to achieve good results in practice.

Theorem 5 ([LS19])**.** *Let $\mathcal{C}$ be the result of k-means++ initialization with $Z \geq 100000k \log\log k$ rounds of local search. Then, $\mathbb{E}[\phi_X(\mathcal{C})] = O(\phi_X(\mathcal{C}_{OPT}))$.*

The key insight to Theorem 5 comes from the following lemma:

Lemma 6 ([LS19])**.** *Let $\mathcal{C}$ be a set of centers with cost $\phi_X(\mathcal{C}) > 500\phi_X(\mathcal{C}_{OPT})$, and let $\mathcal{C}'$ be $\mathcal{C}$ after one round of local search. Then, $\phi_X(\mathcal{C}') \leq (1 - \frac{1}{100k})\phi_X(\mathcal{C})$ with probability at least $\frac{1}{1000}$.*

Finally, we come to the $O(k)$ rounds constant factor approximation.

Theorem 6 ([CGPR20])**.** *Fix $\epsilon \in (0, 1]$ and require $k = \Omega(1/\epsilon^{20})$. Let $\mathcal{C}$ be the result of k-means++ initialization followed by $Z \geq \epsilon k$ rounds of local search. Then, $\phi_X(\mathcal{C}) \leq 10^{30}/\epsilon^3 \cdot \phi_{\mathrm{OPT}}$ with probability at least $1 - \exp(-\Omega(k^{0.1}))$.*

Note that to get a constant factor approximation in close to k rounds, their result requires that k must be extremely large.

References

[AV06] David Arthur and Sergei Vassilvitskii. k-means++: The Advantages of Careful Seeding. 2006.

[BLHK16a] Olivier Bachem, Mario Lucic, S. Hamed Hassani, and Andreas Krause. Approximate k-means++ in sublinear time. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 1459–1467. AAAI Press, 2016.

[BLHK16b] Olivier Bachem, Mario Lucic, S. Hamed Hassani, and Andreas Krause. Fast and provably good seedings for k-means. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 55–63, Red Hook, NY, USA, 2016. Curran Associates Inc.

[BMV$^+$12] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7):622–633, March 2012.

[BR13] Tobias Brunsch and Heiko Röglin. A bad instance for k-means++. *Theoretical Computer Science*, 505:19–26, 2013. Theory and Applications of Models of Computation 2011.

[CGPR20] Davin Choo, Christoph Grunau, Julian Portmann, and Václav Rozhon. k-means++: few more steps yield constant approximation. *CoRR*, abs/2002.07784, 2020.

[DKP20] Amit Deshpande, Praneeth Kacham, and Rameshwar Pratap. Robust k-means++. In Jonas Peters and David Sontag, editors, *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 799–808. PMLR, 03–06 Aug 2020.

[HLP88] G. H. Hardy, John E. Littlewood, and George Pólya. *Inequalities.* Cambridge University Press, Cambridge, 1988.

[Llo82] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[LS19] Silvio Lattanzi and Christian Sohler. A Better k-means++ Algorithm via Local Search. 2019.

[MRS20] Konstantin Makarychev, Aravind Reddy, and Liren Shan. Improved guarantees for k-means++ and k-means++ parallel. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16142–16152. Curran Associates, Inc., 2020.

[Wei16] Dennis Wei. A constant-factor bi-criteria approximation guarantee for k-means++. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.