Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

**Approximate K-Means++ in Sublinear Time**

|  |  |  |  |
| --- | --- | --- | --- |
| **Olivier Bachem** | **Mario Lucic** | **S. Hamed Hassani** | **Andreas Krause** |
| ETH Zurich | ETH Zurich | ETH Zurich | ETH Zurich |
| olivier.bachem@inf.ethz.ch | lucic@inf.ethz.ch | hamed@inf.ethz.ch | krausea@ethz.ch |

**Abstract**

The quality of K-Means clustering is extremely sen- sitive to proper initialization. The classic remedy is to apply k-means++ to obtain an initial set of cen- ters that is provably competitive with the optimal solu- tion. Unfortunately, k-means++ requires k full passes over the data which limits its applicability to massive datasets. We address this problem by proposing a sim- ple and efﬁcient seeding algorithm for K-Means cluster- ing. The main idea is to replace the exact D2 -sampling step ink-means++ with a substantially faster approx- imation based on Markov Chain Monte Carlo sam- pling. We prove that, under natural assumptions on the data, the proposed algorithm retains the full theoreti- cal guarantees of k-means++ while its computational complexity is only sublinear in the number of data points. For such datasets, one can thus obtain a *prov- ably* good clustering in sublinear time. Extensive exper- iments conﬁrm that the proposed method is competi- tive with k-means++ on a variety of real-world, large- scale datasets while offering a reduction in runtime of several orders of magnitude.

**1 Introduction**

The goal of K-Means clustering is to ﬁnd a set of k cluster centers for a dataset such that the sum of squared distances of each point to its closest cluster center is minimized. It is one of the classic clustering problems and has been stud- ied for several decades. Yet even today, it remains a relevant problem: *Lloyd’s algorithm* (Lloyd, 1982), a local search al- gorithm for K-Means, is still one of the ten most popular algorithms for data mining according to Wu et al. (2008) and is implemented as a standard clustering method in most machine learning libraries. In the last few years, K-Means clustering has further been studied in various ﬁelds of ma- chine learning such as representation learning (Coates, Lee, and Ng, 2011; Coates and Ng, 2012) and Bayesian nonpara- metrics (Kulis and Jordan, 2012).

It is well-known that K-Means clustering is highly sen- sitive to proper initialization. The classical remedy is to use a seeding procedure proposed by Arthur and Vassilvitskii (2007) that together with Lloyd’s algorithm is known as

Copyright oc 2016, Association for the Advancement of Artiﬁcial Intelligence (www.aaai.org). All rights reserved.

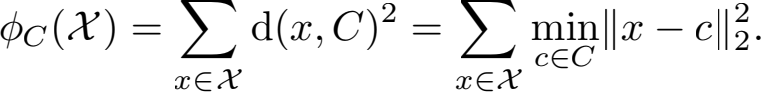
k-means++. In the seeding step of k-means++, the clus- ter centers are sampled iteratively using D2 *-sampling*: First, a cluster center is chosen uniformly at random from the data points. Then, in each of k iterations, a data point is selected as a new cluster center with probability proportional to its distance to the already sampled cluster centers. Even without assumptions on the data, the resulting solution is in expec- tation O(log k)-competitive with regards to the optimal so- lution (Arthur and Vassilvitskii, 2007). While k-means++ is easy to implement, it is non-trivial to apply it to large problems. k-means++ has to make a full pass through the data for every cluster center sampled. This leads to a complexity of Θ(nkd) where n is the number of data points, k the number of cluster centers and d the dimensionality of the data. Even ifk is moderate, this can be computationally infeasible for massive datasets. This motivates our search for a seeding method with a lower, potentially even sub- linear, complexity in the number of data points that retains both the empirical and theoretical beneﬁts of k-means++.

But is it even worth pursuing a fast seeding algorithm? After all, both evaluating the quality of such a seeding and running one iteration of Lloyd’s algorithm exhibit the same Θ(nkd) complexity as the seeding step of k-means++. Hence, one might argue that there is no beneﬁt in reducing the complexity of the k-means++ seeding step as it is dominated by these two other operations. There are two shortcomings to this argument: Firstly, k-means++ is an inherently sequential algorithm of k dependent iterations and, as such,difﬁcult to parallelize in a distributed setting. Evaluating the quality of a K-Means solution, however, can be done in parallel using a single MapReduce step. Similarly, Lloyd’s algorithm can also be implemented in MapReduce (Zhao, Ma, and He, 2009). Secondly, there are many use cases where one requires fast seeding without evaluating the quality of the seeding or running Lloyd’s algorithm subsequently. For example, the quality of such a solution can be improved using fast algorithms such as online (Bottou and Bengio, 1994) or mini-batch K-Means (Sculley, 2010) which may be run for less than O(n) iter- ations in practice. Furthermore, various theoretical results such as coreset constructions (Bachem, Lucic, and Krause, 2015) rely on the theoretical guarantee of k-means++. Hence, a fast seeding algorithm with a strong theoretical guarantee would have an impact on all these applications.

**Our Contributions.** In this paper, we propose a sim- ple, but novel algorithm based on *Markov Chain Monte Carlo* (MCMC) sampling to quickly obtain a seeding for the K-Means clustering problem. The algorithm can be run with varying computational complexity and approximates the seeding step of k-means++ with arbitrary precision as its complexity is increased. Furthermore, we show that for a wide class of non-pathological datasets convergence is fast. Under these mild and natural assumptions, it is sufﬁ- cient to run our algorithm with complexity sublinear in the number of data points to retain the same O(log k) guarantee as k-means++. This implies that for such datasets, a prov- ably good K-Means clustering can be obtained in *sublinear time*. We extensively evaluate the proposed algorithm empir- ically and compare it to k-means++ as well as two other approaches on a variety of datasets.

**2 Background & Related Work**

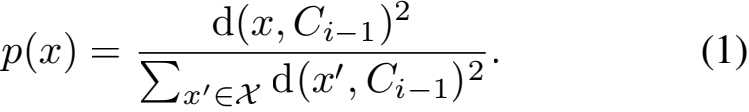
**K-Means clustering.** Let X denote a set of n points in Rd. The *K-Means clustering problem* is to ﬁnd a set C of kclus- ter centers in Rd such that the quantization error φC (X) is minimized, where



In this paper, we implicitly use the Euclidean distance func- tion; however, any distance function d(x, x/ ) may be used.

The optimal quantization error is denoted by φPT (X).

**k-means++ seeding.** The seeding step of k-means++ (Arthur and Vassilvitskii 2007) works by sampling an initial cluster center uniformly at random and then adaptively sam- pling (k − 1) additional cluster centers using D2 *-sampling*. More speciﬁcally, in each iteration i = 2, . . . ,k, the data point x ∈ X is added to the set of already sampled cluster centers Ci−1 with probability



The algorithm’s time complexity is Θ(nkd) and the result- ing seeding Ck is in expectation O(log k) competitive with

respect to the optimal quantization error φPT (X) (Arthur

and Vassilvitskii, 2007), i.e.

E [φCk (X)] ≤ 8(log2 k +2)φPT (X).

**Related work.** Previously, the same idea as in k-means++ was explored in Ostrovsky et al. (2006) where it was shown that, under some data separability assump- tions, the algorithm provides a constant factor approxima- tion. Similar assumptions were analyzed in Balcan, Blum, and Gupta (2009), Braverman et al. (2011), Shindler, Wong, and Meyerson (2011), Jaiswal and Garg (2012) and Agar- wal,Jaiswal, and Pal (2013). Without any assumption on the data,it was shown that D2 -sampling leads to a constant fac- tor approximation if Ω(klog k) (Ailon, Jaiswal, and Mon- teleoni, 2009) or Ω(k) (Aggarwal, Deshpande, and Kannan, 2009) centers are sampled. Bad instances for k-means++

were considered in the original paper (Arthur and Vassilvit- skii, 2007) as well as in Brunsch and Rglin (2011). A poly- nomial time approximation scheme for K-Means using D2 - sampling was proposed in Jaiswal, Kumar, and Sen (2014) and Jaiswal, Kumar, and Yadav (2015).

Several ideas extending k-means++ to the streaming setting were explored: A single-pass streaming algorithm based on coresets and k-means++ was proposed in Ack- ermann et al. (2012). The main drawback of this approach is that the size of the coreset is exponential in the dimen- sionality of the data. Ailon, Jaiswal, and Monteleoni (2009) suggest a streaming algorithm based on Guha et al. (2003) that provides the same O(log k) guarantee as k-means++ with a complexity of O(ndk log n log k).

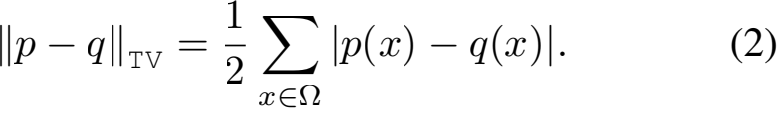
Bahmani et al. (2012) propose a parallel version of k-means++ called k-meansⅡ that obtains the same O(log k) guarantee with a complexity of Θ(ndk log n). The main idea is to replace the k sequential sampling rounds of k-means++ by r = Θ(log n) rounds in each of which l = Θ(k) points are sampled in parallel. In a ﬁnal step, the Θ(klog n) sampled points are clustered again using k-means++ to produce a ﬁnal seeding of k points. As a result, the computational complexity of k-meansⅡ is higher than k-means++ but can be efﬁciently distributed across different machines. In Section 6, we will compare k-meansⅡ with our proposed method on various datasets.

**3 Approximate D**2 **-sampling**

In each iteration of D2 -sampling, the k-means++ algo- rithm has a computational complexity of Θ(nd) as it needs to calculate the sampling probabilities p(x) in (1) for ev- ery data point. We aim to reduce the complexity by approx- imating the D2 -sampling step: we strive for a fast *sampling*

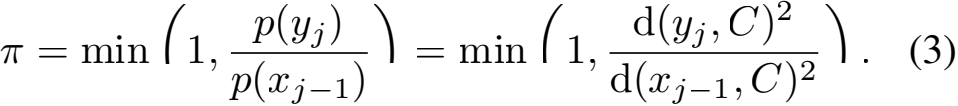
*scheme* whose implied sampling probabilities (x) are close

top(x). To formalize this notion of closeness, we use the *to- tal variation distance* which measures the maximum differ- ence in probabilities that two distributions assign to an event. More formally, let Ω be a ﬁnite sample space on which two probability distributions p and q are deﬁned. The total vari- ation distance between p and q is given by



In Section 5 we will show that using total variation distance we can bound the solution quality obtained by our algo- rithm. Informally, if the total variation distance is less than ∈, we are able to to retain the same theoretical guarantees as k-means++ with probability at least (1 − ∈).

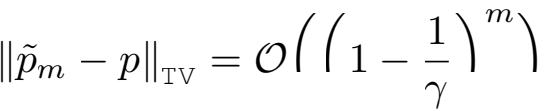
**MCMC approximation.** The *Metropolis-Hastings algo- rithm* (Hastings 1970) (with an independent, uniform pro- posal distribution) applied to a single step of D2 -sampling works as follows: We uniformly sample an initial state x0 from the point set X and then iteratively build a Markov chain. In each iteration j, we uniformly sample a candidate pointyj and calculate the acceptance probability



With probability π we then set the state xj to yj and with probability 1 −π to xj−1 . For a Markov chain of total length m, we only need to calculate the distance between m data points and the cluster centers since the normalization con- stants of p(yj ) and p(xj−1) in (3) cancel. By design, the stationary distribution of this Markov chain is the target dis-

tribution p(x). This implies that the distribution m (x) of

the m-th state xm converges to p(x) as m → ∞ . Further- more, the total variation distance decreases at a geometric rate with respect to the chain length m (Cai, 2000) as



where

γ = n p(x). (4)

This implies that there is a chain length m = O 

that achieves a total variation distance of at most ∈ . Intu-

itively, γ measures the difﬁculty of approximately sampling

from p(x) and depends on the current set of centers C and

the dataset X. We remark that the total variation distance in-

creases with γ . For now, we assume γ to be given and defer

the detailed analysis to Section 5.

**4 Approximate K-Means++ using K-MC**2

It is straightforward to extend this MCMC-based sampler to

approximate the full seeding step of k-means++: We ﬁrst

sample an initial cluster center uniformly at random. Then,

for each of the remaining k − 1 iterations, we build an inde-

pendent Markov chain of length m and use the last element

as the new cluster center. We call this algorithm K-MC2 and

provide pseudo-code in Algorithm 1. The complexity of the

proposed algorithm is Θ (mk2 d). In particular, it does not

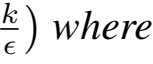
depend on the number of data points n.

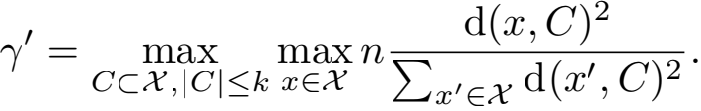
Theorem 1 guarantees convergence of Algorithm 1 to k-means++ in terms of total variation distance. Since the (k − 1) Markov chains are independent, we may use a union bound: If the sampling induced by each chain has a total variation distance of at most ∈/(k − 1), then the total vari- ation distance between the sampling induced by K-MC2 and the sampling induced by k-means++ is at most ∈ (as shown in the proof of Theorem 1).

|  |
| --- |
| **Algorithm 1** K-MC2 |
| **Require:** Dataset X, number of centers k, chain length m c1 ← point uniformly sampled from X  C1 ← {c1 }  **for** i = 2, 3, . . . ,k **do**  x ← point uniformly sampled from X dx ← d(x, Ci−1)2  **for** j = 2, 3, . . . ,m **do**  y ← point uniformly sampled from X dy ← d(y, Ci−1)2  **if** > Unif(0, 1) **then**  x ← y, dx ← dy Ci ← Ci−1 ∪{x}  **return** Ck |

**Theorem 1.** *Let* k > 0 *and* 0 < ∈ < 1*. Let* p++ (C) *be the probability of sampling a seeding* C *using* k-means++ *and* pmcmc (C) *the probability using* K-MC2 *(Algorithm 1). Then,*

Ⅱpmcmc − p++ Ⅱ TV ≤ ∈

*for a chain length* m = O(γ/ log 

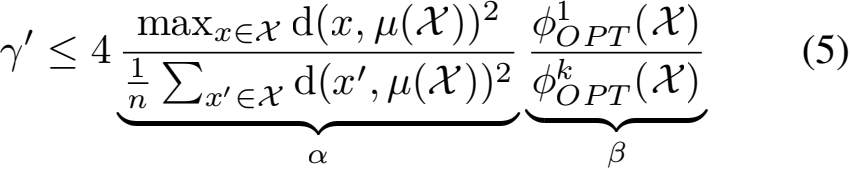


*The resulting complexity of Algorithm 1 is* O(γ/ k2 d log *.*  The proof is given in Section B of the Appendix. This re- sult implies that we can use K-MC2 to approximate the seed- ing step of k-means++ to arbitrary precision. The required chain length m depends linearly on γ/ which is a uniform upper bound on γ for all possible sets of centers C. In the next section, we provide a detailed analysis of γ/ and quan- tify its impact on the quality of seeding produced by K-MC2 .

**5 Analysis**

In the previous section, we saw that the rate of convergence of K-MC2 depends linearly on γ/ . By deﬁnition, γ/ is trivially bounded by n and it is easy to construct a dataset achieving this bound: Consider the 2-Means clustering problem and let (n − 1) points be in an arbitrarily small cluster while a single point lies at some distance away. With probability (1 −  - tial cluster center. In the subsequent D2 -sampling step, we are thus required to sample the single point with probabil- ity approaching one. For such a pathological dataset, it is impossible to approximate D2 -sampling in sublinear time. Our proposed algorithm is consistent with this result as it would require linear complexity with regards to the number of data points for this dataset. Fortunately, such pathological datasets rarely occur in a practical setting. In fact, under very mild and natural assumptions on the dataset, we will show that γ/ is at most sublinear in the number of data points.

To this end, we assume that the dataset X is sampled i.i.d. from a base distribution F and note that γ/ can be bounded by two terms α and β, i.e.



where μ(X) denotes the mean of X and φPT (X) denotes

the quantization error of the optimal solution of k centers (see Section C of the Appendix for a proof).

**Tail behavior of distribution** F**.** The ﬁrst term α mea- sures the ratio between the maximum and the average of the squared distances between the data points and their empiri- cal mean. In the pathological example introduced above, α would approach (n − 1). Yet, under the following assump- tion, α grows sublinearly in n as formally stated and proven in Section A.1 of the Appendix.

**(A1)** For distributions F with ﬁnite variance and exponen- tial tails1 , α is independent of k and d and w.h.p.

α = O(log2 n). ~

1 ∃c,t such that P [d(x, μ(F)) > a] ≤ ce −at where x F.

This assumption is satisﬁed by the univariate and multivari- ate Gaussian as well as the Exponential and Laplace dis- tributions,but not by heavy tailed distributions such as the Pareto distribution. Furthermore, if α is sublinear in n for all components of a mixture, then α is also sublinear for the mixture itself. For distributions with ﬁnite variance and bounded support, we even show a bound on α that is inde- pendent of n.

**Nondegeneracy of distribution** F**.** The second term β measures the reduction in quantization error ifk centers are used instead of just one. Without prior assumptions β can be unbounded: If a dataset consists of at most k distinct points, the denominator of the second term in (5) is zero. Yet, what is the point of clustering such a dataset in practice if the solution is trivial? It is thus natural to assume that F is non- degenerate,i.e., its support is larger thank. Furthermore, we expect β to be independent of n if n is sufﬁciently large: Due to the strong consistency of K-Means the optimal solution on a ﬁnite sample converges to the optimal quantizerof the gen- erating distribution as n → ∞ (Pollard, 1981) and such an optimal quantizeris by deﬁnition independent of n. At the same time, β should be non-increasing with respect to k as additional available cluster centers can only reduce the opti- mal quantization error. This allows us to derive a very gen- eral result (formally stated and proven in Section A.2 of the Appendix) that for distributions F that are “approximately uniform” on a hypersphere, β is independent of n.

**(A2)** For distributions F whose minimal and maximal den- sity on a hypersphere with nonzero probability mass is bounded by a constant, β is independent of n and w.h.p.

β = O(k).

This property holds for a wide family of continuous prob- ability distribution functions including the univariate and multivariate Gaussian, the Exponential and the Laplace dis- tribution. Again, if β is bounded for all components of a mixture, then β is also bounded for the mixture.

**Solution quality of K-MC**2**.** These two assumptions do not only allow us to bound γ/ and thus obtain favourable convergence, but also to analyze the quality of solutions gen- erated by K-MC2 . In particular, we show in Section C of the Appendix that the expected quantization error φK -MC 2 of Al- gorithm 1 is bounded by

E [φK -MC 2 ] ≤ E [φk-means++]+2∈βφPT (X).

Hence, by setting the total variation distance ∈ = O(1/β),

the second term becomes a constant factor of φPT (X). By

applying Theorem 1 with m = O(αβ log βk), the solution sampled from K-MC2 is in expectation O(log k)-competitive to the optimal solution and we obtain the following theorem. **Theorem 2.** *Let* k > 0 *and* X *be a dataset with* α = O(log2 n) *and* β = O(k)*, i.e. assume* ***(A1)*** *and* ***(A2)****. Let* C *be the set of centers sampled by* K-MC2 *(Algorithm 1) with* m = O(k log2 nlog k)*. Then we have*

E [φC (X)] ≤ O(log k)φPT (X).

*The total complexity is* O(k3 d log2 nlog k)*.*

Table 1: Datasets with size n, dimensionality d and esti- mated values for α and β

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DATASET | N | D | α | K=200) |
| CSN | 80000 | 17 | 546.27 | 3.04 |
| KDD | 145751 | 74 | 1267.65 | 1.81 |
| USGS | 59209 | 3 | 2.68 | 51.67 |
| WEB | 45811883 | 5 | 2.33 | 57.09 |
| BIGX | 11620300 | 57 | 7.82 | 14.17 |
| SONG | 515345 | 90 | 525.67 | 1.23 |

The proof is provided in Section C of the Appendix. The signiﬁcance of this result is that, under natural assumptions, it is sufﬁcient to run K-MC2 with complexity sublinear in the number of data points to retain the theoretical guarantee of k-means++. Hence, one can obtain a *provably* good clus- tering for K-Means in sublinear time for such datasets.

**6 Experiments**

**Datasets.** We use six different datasets: USGS (United States Geological Survey, 2010), CSN (Faulkner et al., 2011), KDD (KDD Cup, 2004), BIGX (Ackermann et al., 2012), WEB (Yahoo! Labs, 2008) and SONG (Bertin-Mahieux et al., 2011). Table 1 shows the size and number of dimensions of these datasets as well as estimates of both α and β . We directly calculate α using (5) and approximate β by replac-

ing the optimal solution φPT (X) in (5) with the solution

obtained using k-means++.

**Methods.** We compare the algorithm K-MC2 to four alter- native methods (k-means++, RANDOM, HEURISTIC and k-meansⅡ). We run K-MC2 with different chain lengths, i.e. m ∈ {1, 2, 5, 10, 20, 50, 100, 150, 200}. As the main baselines, we consider the seeding step of k-means++ as well as RANDOM, a seeding procedure that uniformly samples k data points as cluster centers. We further pro- pose the following HEURISTIC : It works by uniformly sampling s points and then running the seeding step of k-means++ on this subset. Similar to K-MC2 , we set s ∈ {100, 200, 500, . . . , 10/ 000, 15/ 000, 20/ 000}. Fi- nally, we also compare to k-meansⅡ . We use r = 5 rounds and a variety of oversampling factors, i.e. l ∈ {0.02k, 0.05k, 0.1k, 0.2k, 0.5k, 1k, 2k}.

**Experimental setup.** For the datasets USGS, CSN and KDD, we set k = 200 and train all methods on the full datasets. We measure the number of distance evaluations and the quality of the solution found in terms of quantiza- tion error on the full dataset. For the datasets BIGX, WEB and SONG, we set k = 2000 and train on all but 250/ 000 points which we use as a holdout set for evaluation. We con- sider both training error and holdout error for the following reason: On one hand, the theoretical guarantees for both K - MC2 and k-means++ hold in terms of training error. On the other hand, in practice, one is usually interested in the generalization error.

As all the considered methods are randomized proce- dures, we run them repeatedly with different initial random seeds. We average the obtained quantization errors and use

Table 2: Experimental results.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | RELATIVE ERROR VS . K-MEANS++ | | | | | | SPEEDUP VS . K-MEANS++ (DISTANCE EVALUATIONS)  CSN KDD USGS BIGX WEB SONG |
| CSN | KDD | USGS | BIGX | WEB | SONG |
| K-MEANS++ | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 1.0 × 1.0 × 1.0 × 1.0 × 1.0 × 1.0 × |
| RANDOM | 394.50% | 307.44% | 315.50% | 11.45% | 105.34% | 9.74% | - - - - - - |
| K-MC2 (m = 20) | 63.58% | 32.62% | 2.63% | 0.05% | 0.77% | 0.38% | 40.0 × 72.9 × 29.6 × 568.5 × 2278.1 × 13.3 × |
| K-MC2 (m = 100) | 14.67% | 2.94% | -0.33% | 0.13% | -0.00% | -0.02% | 8.0 × 14.6 × 5.9 × 113.7 × 455.6 × 2.7 × |
| K-MC2 (m = 200) | 6.53% | 1.00% | -0.83% | -0.03% | 0.01% | -0.02% | 4.0 × 7.3 × 3.0 × 56.9 × 227.8 × 1.3 × |
| HEURISTIC (s = 2000) | 94.72% | 73.28% | 5.56% | 0.38% | 2.12% | 0.69% | 40.0 × 72.9 × 29.6 × 568.5 × 2278.1 × 13.3 × |
| HEURISTIC (s = 10000) | 29.22% | 9.55% | 0.20% | 0.10% | 0.15% | 0.15% | 8.0 × 14.6 × 5.9 × 113.7 × 455.6 × 2.7 × |
| HEURISTIC (s = 20000) | 13.99% | 2.22% | 0.27% | 0.02% | 0.07% | 0.05% | 4.0 × 7.3 × 3.0 × 56.9 × 227.8 × 1.3 × |
| K-MEANS Ⅱ (r = 5, l = 0 .02k) | 335.61% | 118.03% | 2356.06% | 223.43% | 562.23% | 40.54% | 9.6 × 9.0 × 8.9 × 10.0 × 9.5 × 9.8 × |
| K-MEANS Ⅱ (r = 5, l = 0 .2k) | 2.12% | 0.71% | 19.13% | 1.74% | 11.03% | -0.34% | 1.0 × 1.0 × 1.0 × 1.0 × 1.0 × 1.0 × |
| K-MEANS Ⅱ (r = 5, l = 2k) | -3.75% | -6.22% | -3.78% | -2.43% | -2.04% | -5.16% | 0.1 × 0.1 × 0.1 × 0.1 × 0.1 × 0.1 × |

the standard error of the mean to construct 95% conﬁdence intervals. For each method, we further calculate the relative error and the speedup in terms of distance evaluations with respect to our main baseline k-means++.

**Discussion.** The experimental results are displayed in Figures 1 and 2 and Table 2. As expected, k-means++ produces substantially better solutions than RANDOM (see Figure 1). For m = 1, K-MC2 essentially returns a uniform sample of data points and should thus exhibit the same solu- tion quality as RANDOM. This is conﬁrmed by the results in Figure 1. As the chain length m increases, the performance of K-MC2 improves and converges to that of k-means++. Even for small chain lengths, K-MC2 is already competitive with the full k-means++ algorithm. For example, on BIGX, K-MC2 with a chain length of m = 20 exhibits a relative er- ror of only 0.05% compared to k-means++ (see Table 2). At the same time, K-MC2 is 586.5 × faster in terms of dis- tance evaluations.

K-MC2 signiﬁcantly outperforms HEURISTIC on all datasets (see Figure 1). For the same number of distance evaluations K-MC2 achieves a smaller quantization error: In the case of BIGX, HEURISTIC with s = 2000 exhibits a rel- ative error of 0.38% compared to the 0.05% of K-MC2 with a chain length of m = 20. In contrast to HEURISTIC, K-MC2 further offers the theoretical guarantees presented in Theo- rems 1 and 2.

Figure 2 shows the relationship between the performance of k-meansⅡ and the number of distance evaluations. Even with ﬁve rounds, k-meansⅡ is able to match the perfor- mance of the inherently sequential k-means++ and even outperforms it if more computational effort is invested. However, as noted in the original paper (Bahmani et al., 2012),k-meansⅡ performs poorly if it is run with low com- putational complexity, i.e. ifr · l < k.

As such, K-MC2 and k-meansⅡ have different use sce- narios: k-meansⅡ allows one to run the full k-means++ seeding step in a distributed manner on a cluster and poten- tially obtain even better seedings than k-means++ at the cost computational effort. In contrast, K-MC2 produces ap- proximate but competitive seedings on a single machine at a fraction of the computational cost of both k-means++ and k-meansⅡ .

**7 Conclusion**

We propose K-MC2 , an algorithm to quickly obtain an ini- tial solution to the K-Means clustering problem. It has sev- eral attractive properties: It can be used to approximate the seeding step of k-means++ to arbitrary precision and, un- der natural assumptions, it even obtains *provably* good clus- terings in sublinear time. This is conﬁrmed by experiments on real-world datasets where the quality of produced clus- terings is similar to those of k-means++ but the runtime is drastically reduced. K-MC2 further outperforms a heuris- tic approach based on subsampling the data and produces fast but competitive seedings with a computational budget unattainable by k-meansⅡ . We posit that our technique can be extended to improve on other theoretical results for D2 - sampling as well as to other clustering problems.

**Acknowledgments.** We would like to thank Sebastian Tschiatschek and the anonymous reviewers for their com- ments. This research was partially supported by ERC StG 307036 and the Zurich Information Security Center.

**References**

Ackermann, M. R.; Mrtens, M.; Raupach, C.; Swierkot, K.; Lam- mersen, C.; and Sohler, C. 2012. StreamKM++: A clustering al- gorithm for data streams. *Journal of Experimental Algorithmics* 17:2–4.

Agarwal, M.; Jaiswal, R.; and Pal, A. 2013. k-means++ under approximation stability. In *Theory and Applications of Models of Computation*. Springer. 84–95.

Aggarwal, A.; Deshpande, A.; and Kannan, R. 2009. Adaptive sampling for k-means clustering. In *Approximation, Randomiza- tion, and Combinatorial Optimization. Algorithms and Techniques*. Springer. 15–28.

Ailon, N.; Jaiswal, R.; and Monteleoni, C. 2009. Streaming k- means approximation. In *Advances in Neural Information Pro- cessing Systems (NIPS)*, 10–18.

Arthur, D., and Vassilvitskii, S. 2007. k-means++: The advantages of careful seeding. In *Symposium on Discrete Algorithms (SODA)*, 1027–1035. Society for Industrial and Applied Mathematics.

Bachem, O.; Lucic, M.; and Krause, A. 2015. Coresets for non- parametric estimation - the case of DP-means. In *International Conference on Machine Learning (ICML)*.

×1011 KDD (k=200)

USGS (k=200)

×105 CSN (k=200)

× 102

7

9

8

7

6

5

4

3

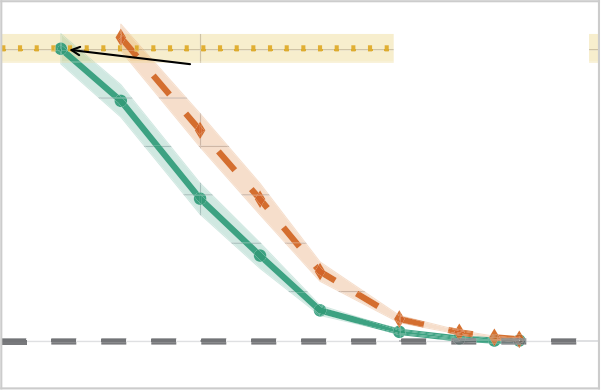
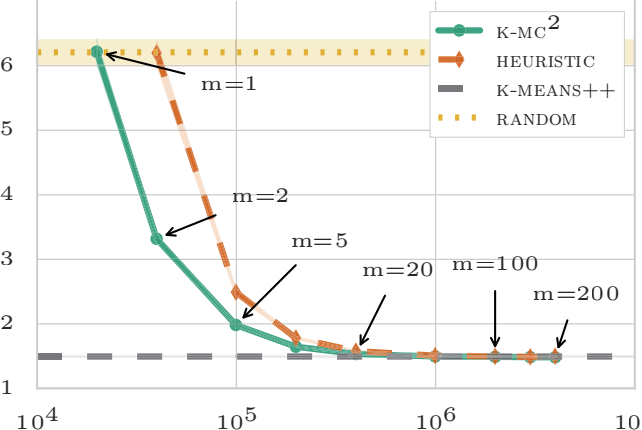
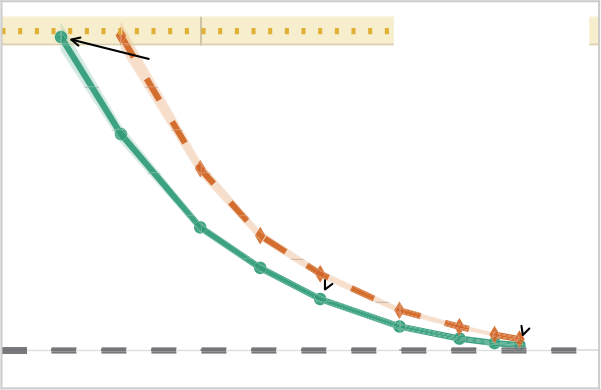
2

1

5 .0 4 .5 4 .0 3 .5 3 .0 2 .5 2 .0 1 .5 1 .0 0 .5

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | k-mc2 |
|  |  | | heuristic |
|  |  | | k-means++  random |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  |  |  |
|  |  | |  |
|  |  | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | k-mc2 | |
|  |  | | heuristic  k-means++ | |
|  |  | | random | |
|  |  | |  | |
|  |  | |  | |
|  |  |  |  | |
|  |  | |  |  |
|  |  | |  | |



Holdout error Training error

m=1

m=2

m=5

m=20

m=100

m=200

|  |
| --- |
|  |

|  |
| --- |
|  |

m=1

m=2

m=5

m=20

m=100

m=200

7

104 105 106 107

Distance evaluations

104 105 106 107

Distance evaluations

Distance evaluations

×1011 SONG (k=2000)

×1010 BIGX (k=2000)

×102 WEB (k=2000)

7 .3

7 .2

7 . 1

7 .0

6 .9

6 .8

6 .7

6 .6

6 .5

2 .2

1 .34

1 .32

1 .30

1 .28

1 .26

1 .24

1 .22

1 .20

1 . 18

m=1

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  | | | | |  | | | | |
|  | |  | | | | |  | | | | |
|  | |  | | | | |  | | | | |
|  | |  | | | | |  | | | | |
|  |  | m=2 | | | | |  | | | | |
|  | | m=5 | | | | | m=100 | | | | |
|  | |  | m=20 | | | |  | | m=200 | | |
|  |
|  | |  | |  |  |  |  |  |  |  |  |
|  | |  | | | | |  | | | | |

m=1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | |  | | k-mc2 | | |
|  | |  | | heuristic  k-means++ | | |
|  | |  | | random | | |
|  | | m=2 | |  | | |
|  | | m=5 | |  | | |
|  |  |  |  | m=100  m=200 | | |
|  |
|  | |  | |  |  |  |
|  | |  | |  | | |

|  |
| --- |
|  |

|  |
| --- |
|  |

2 .0

1 .8

1 .6

1 .4

m=20

1 .2

m=200

1 .0

0 .8

 k-mc2 

 heuristic

 k-means++

 random

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | k-mc2   |  | | --- | |  | | |  |
| m=1 |  | heuristic   k-means++ | | |
|  |  | random | | |
|  | m=2 |  | | |
|  | m=5  m=20 | m=100 | | |
|  |  |  |  | |
|  |  |  | | |

106 107 108 109

106 107 108 109

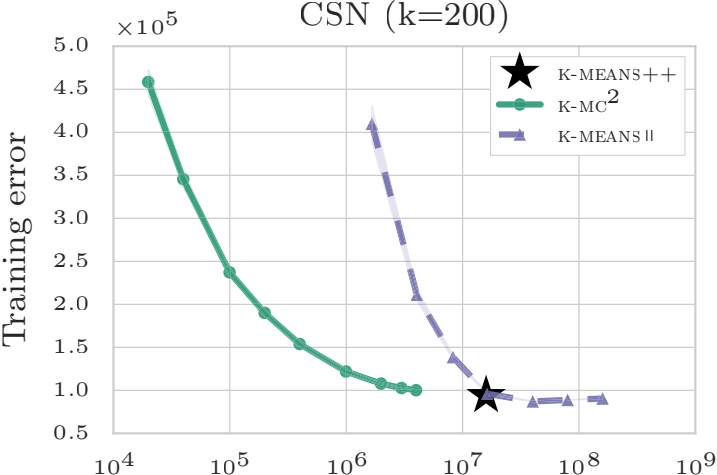
Distance evaluations

106 107 108 109

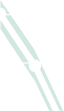
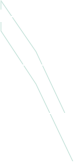
Distance evaluations

Distance evaluations

Figure 1: Average quantization error vs. number of distance evaluations for K-MC2 and HEURISTIC as well as the average quantization error (without the number of distance evaluations) fork-means++ and RANDOM. K-MC2 quickly converges to full k-means++ and outperforms HEURISTIC. Shaded areas denote 95% conﬁdence intervals.

×1011 KDD (k=200)

×103 USGS (k=200)

9

4 .0

3 .5

3 .0

2 .5

2 .0

1 .5

1 .0

0 .5

0 .0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k- | | | means++ |
|  |
|  |  |  | |  | | k- k- | mc2  means Ⅱ |
|  |  |  | |  | | |  |
|  |  |  | |  | | |  |
|  |  |  | |  | | |  |
|  |  |  | |  | | |  |
|  |  |  | |  | | |  |
|  |  |  | |  | | |  |
|  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | |  |  | k- | means++ |
|  |  | |  |  | k- k- | mc2  means Ⅱ |
|  | |  |  |  |  |  |
|  | | |  |  |  |  |
|  | | |  |  |  |  |
|  | | |  |  |  |  |
|  | | |  |  |  |  |
|  | | |  |  |  |  |

8

|  |
| --- |
|  |

|  |
| --- |
|  |

7

6

5

4

3

2



1

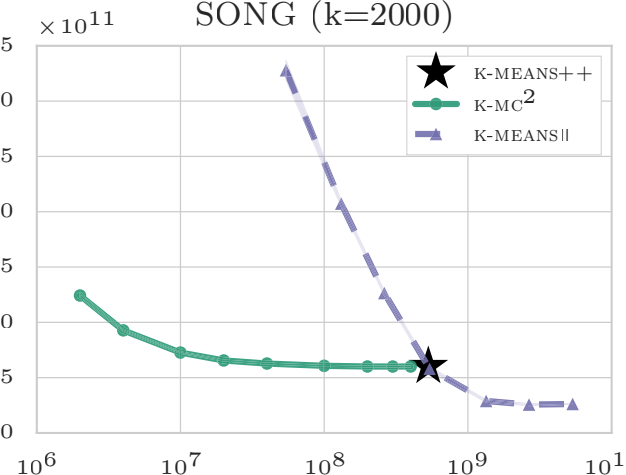
104 105 106 107 108 109

104 105 106 107 108 109

Distance evaluations

Distance evaluations

Distance evaluations

×1010 BIGX (k=2000)

×102 WEB (k=2000)

9 .

7

6

5

4

3

2

4 .0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | k-m k-mc | eans++ 2 |
|  |  |  |  | | k-m | eans Ⅱ |
|  |  |  |  | |  |  |
|  |  |  |  | |  |  |
|  |  |  |  | |  |  |
|  |  |  |  | |  |  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | k-m | eans++ 2 |
|  |  |  |  | k-mc k-m | eans Ⅱ |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |



9 .

Holdout error

|  |
| --- |
|  |

|  |
| --- |
|  |

3 .5

8 .

3 .0

8 .

2 .5

7 .

2 .0

7 .

6 .

1

1 .5



6 .

0

1 .0

0

106 107 108 109 1010 1011 1012

106 107 108 109 1010 1011 1012

Distance evaluations Distance evaluations Distance evaluations

Figure 2: Average quantization error vs. number of distance evaluations for K-MC2 , k-means++ and k-meansⅡ . K-MC2 obtains competitive solutions signiﬁcantly faster than both k-means++ and k-meansⅡ .

1464

Bahmani, B.; Moseley, B.; Vattani, A.; Kumar, R.; and Vassilvit- skii, S. 2012. Scalable k-means++. *Very Large DataBases (VLDB)* 5(7):622–633.

Balcan, M.-F.; Blum, A.; and Gupta, A. 2009. Approximate clus- tering without the approximation. In *Symposium on Discrete Al- gorithms (SODA)*, 1068–1077. Society for Industrial and Applied Mathematics.

Bertin-Mahieux,T.; Ellis,D. P.; Whitman,B.; and Lamere,P. 2011. The Million Song Dataset. In *International Conference on Music Information Retrieval*.

Bottou, L., and Bengio, Y. 1994. Convergence properties of the k- means algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, 585–592.

Braverman, V.; Meyerson, A.; Ostrovsky, R.; Roytman, A.; Shindler, M.; and Tagiku, B. 2011. Streaming k-means on well- clusterable data. In *Symposium on Discrete Algorithms (SODA)*, 26–40. Society for Industrial and Applied Mathematics.

Brunsch, T., and Rglin, H. 2011. A bad instance fork-means++. In *Theory and Applications of Models of Computation*. Springer. 344–352.

Cai, H. 2000. Exact bound for the convergence of Metropolis chains. *Stochastic Analysis and Applications* 18(1):63–71.

Coates, A., and Ng, A. Y. 2012. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*. Springer. 561–580.

Coates, A.; Lee, H.; and Ng, A. Y. 2011. An analysis of single- layer networks in unsupervised feature learning. In *International Conference on Artiﬁcial Intelligence and Statistics (AISTATS)*, vol- ume 1001.

Faulkner, M.; Olson, M.; Chandy, R.; Krause, J.; Chandy, K. M.; and Krause, A. 2011. The next big one: Detecting earthquakes and other rare events from community-based sensors. In *ACM/IEEE In- ternational Conference on Information Processing in Sensor Net- works*.

Guha, S.; Meyerson, A.; Mishra, N.; Motwani, R.; and O’Callaghan, L. 2003. Clustering data streams: Theory and prac- tice. *IEEE Transactions on Knowledge and Data Engineering* 15(3):515–528.

Hastings, W. K. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57(1):97–109. Jaiswal, R., and Garg, N. 2012. Analysis of k-means++ for sepa- rable data. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer. 591–602.

tails, μ is well deﬁned and Ex F [(d(x, μ)] < ∞ . As a re- sult, by the strong law of large numbers, we have almost

Jaiswal, R.; Kumar, A.; and Sen, S. 2014. A simple D2 -sampling based PTAS fork-means and other clustering problems. *Algorith- mica* 70(1):22–46.

Jaiswal, R.; Kumar, M.; and Yadav, P. 2015. Improved analysis of D2 -sampling based PTAS for k-means and other clustering prob- lems. *Information Processing Letters* 115(2):100–103.

KDD Cup. 2004. Protein Homology Dataset. Retrieved from osmot.cs.cornell.edu/kddcup.

Kulis, B., and Jordan, M. I. 2012. Revisiting k-means: New algo- rithms via Bayesian nonparametrics. In *International Conference on Machine Learning (ICML)*, 513–520.

Lloyd, S. 1982. Least squares quantization in PCM. *IEEE Trans- actions on Information Theory* 28(2):129–137.

Ostrovsky, R.; Rabani, Y.; Schulman, L. J.; and Swamy, C. 2006. The effectiveness of Lloyd-type methods for the k-means problem. In *Foundations of Computer Science (FOCS)*, 165–176. IEEE.

Pollard, D. 1981. Strong consistency of k-means clustering. *The Annals of Statistics* 9(1):135–140.

Sculley, D. 2010. Web-scale k-means clustering. In *World Wide Web Conference (WWW)*, 1177–1178. ACM.

Shindler, M.; Wong, A.; and Meyerson, A. W. 2011. Fast and accurate k-means for large datasets. In *NIPS*, 2375–2383.

United States Geological Survey. 2010. Global Earthquakes (1.1.1972-19.3.2010). Retrieved from the mldata.org repository. Wu, X.; Kumar, V.; Ross Quinlan, J.; Ghosh, J.; Yang, Q.; Motoda,

H.; McLachlan, G.; Ng,A.; Liu, B.; Yu, P.; Zhou,Z.-H.; Steinbach, M.; Hand, D.; and Steinberg, D. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14(1):1–37.

Yahoo! Labs. 2008. R6A - Yahoo! Front Page Today Module User Click Log Dataset. Retrieved from research.yahoo.com repository. Zhao, W.; Ma, H.; and He, Q. 2009. Parallel k-means clustering based on MapReduce. In *Cloud Computing*. Springer. 674–679.

**A Formal Statement of Natural Assumptions**

We state the theorems related to the assumptions introduced in Section 5 and provide the corresponding proofs.

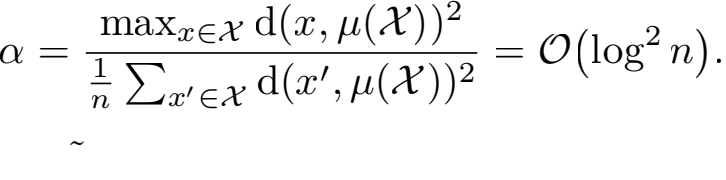
**A.1 Tail behavior of** F

The following theorem corresponds to Assumption **(A1)** in Section 5.

**Theorem 3.** *Let* F *be a probability distribution over* Rd *with ﬁnite variance that has at most exponential tails, i.e.* ∃ c,t*such that*

P [d(x, μ) > a] ≤ ce −at.

*Let* X *be a set of* n *points independently sampled from* F*. Then, with high probability, for* n *sufﬁciently large,* α *is in- dependent of* k *as well as* d *and depends polylogarithmically on* n*, i.e.*



*Proof.*˜ Let μ =  xdF (x~). Since has exponential

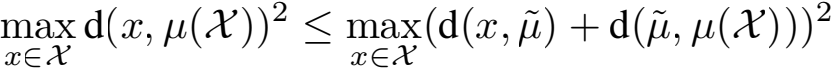
surely that μ(X) → , or d(μ(X), ) → 0 as n → ∞ . Fur-

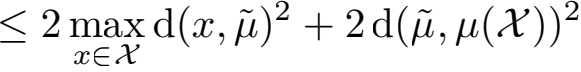
thermore, since F has at most exponential tails P[d(x, ) >

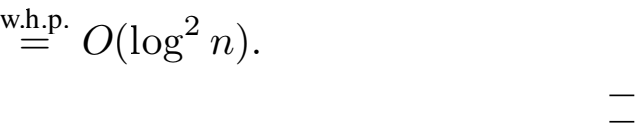
(2 ln n +ln c)/t] ≤ n−2 . Therefore, using the union bound,

with probability at least 1 − 1/n we have that ∀x ∈ X d(x, ) ≤ (2 ln n +ln c)/t.

Hence, maxx∈X d(x, )2 = O(log2 n). Applying the trian- gle inequality, we obtain that







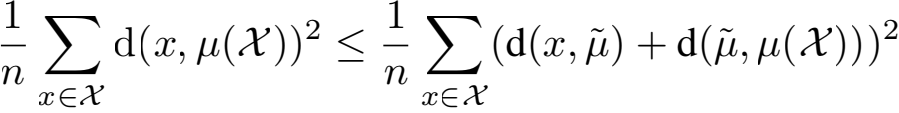
If F has ﬁnite variance and bounded support, we can ob- tain a *constant* bound for α which is formalized by the fol- lowing theorem.

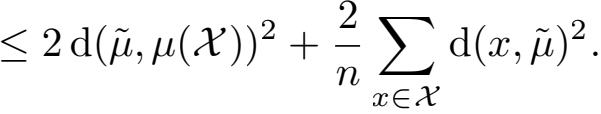
**Theorem 4.** *Let* F *be a probability distribution over* Rd *with ﬁnite variance whose support is almost-surely bounded by a* d*-dimensional sphere with radius* R*. Let* X *be a set of* n *points independently sampled from* F*. Then, with high probability, if* n *is sufﬁciently large,* α *is independent of* n*,* k *and* d*.*

*Proof.* The distance between any point x ∈ X and the mean μ(X) is clearly bounded by 2R. Hence, we always have

maxx∈X d(x, μ(X˜))2 ≤ 4R2 . Also, let =  xdF(x)

and σ2 =  d(x, μ)2 F(x). By using the triangle inequality, we get





Then, by the strong law of large numbers (note that F has a bounded variance), as n grows large, we have almost surely

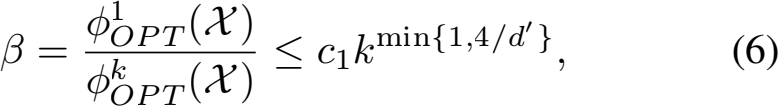
that μ(X) → and 1/nΣx∈X d(x,)2 → σ 2 which con-

cludes the proof.

**A.2 Nondegeneracy of**F

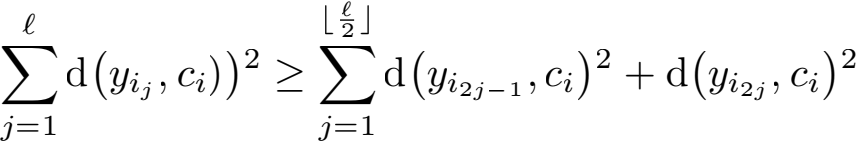
The following theorem corresponds to Assumption **(A2)** in Section 5.

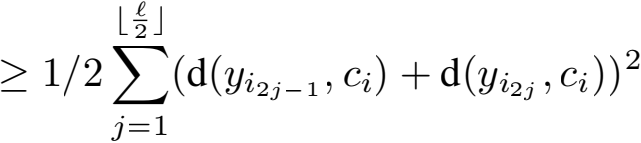
**Theorem 5.** *Let* F *be a probability distribution over* Rd *with ﬁnite variance. Assume that there exists a* d/*- dimensional sphere* B *with radius* R*, s.t.* d/ ≥ 2*,* F(B) > 0*, and* ∀x, y ∈ B : F(x) ≤ cF(y) *for some* c ≥ 1 *(*F *is sufﬁcientlynon-degenerate). Then,w.h.p.*

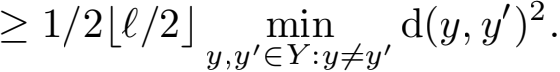


*where* c1 *is a constant inversely proportional to* cF(B)R2*.*

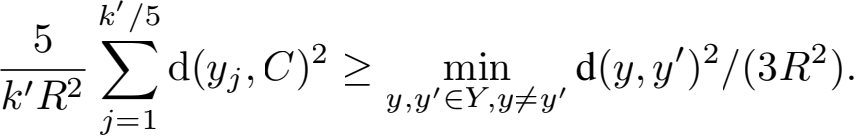
*Proof.* Consider picking n i.i.d. points according to distri- bution F. Among such points, w.h.p m 纟 nF(B)/2 points fall into B. Note that these m points are i.i.d. samples from B according to distribution (x) = F(x)/F(B). Partition these points intom/k/ subsets of size k/ = 15k. Each such subset is also an i.i.d. sample from B according to . Con- sider one of the partitions X = {x1 , ··· ,xk, } and let Y be a randomly chosen subset of X of size k//5. Let C = {c1 , c2 , ··· , ck } ⊂ Rd be an arbitrary set of k centers and assume that for center ci there are l points yi1 , ··· ,yil ∈ Y which have ci as their nearest neighbor. We can then write using the triangle inequality



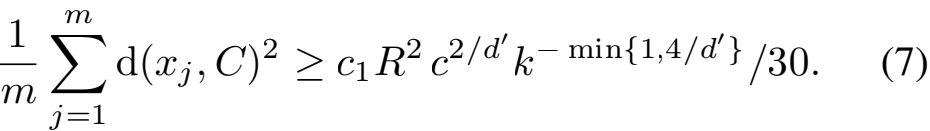




By summing over all the centers, we obtain that



Recall that we have partitioned the m points into m/k/ groups of k/ points. By applying Lemma 1 (see below) and Hoeffding’s inequality, with high probability we have that



Since F has bounded variance then w.h.p. φPT (X)/n con- verges to the variance ofF. Hence, by (7), we have w.h.p.

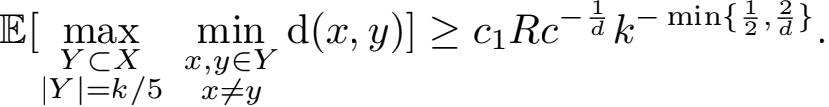
φPT (X)/n ≥ k− min{1 ,4/d, } (c1 R2 F(B)c2/d, )/30.

We conclude that w.h.p. β ≤ c2 R2 F(B)c2/d, kmin{1 ,4/d, }.



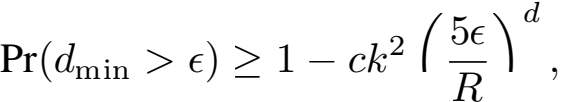
**Lemma 1.** *Let* F *be a probability distribution deﬁned on a* d > 2*-dimensional sphere* B *with radius* R*. Assume that for any two points* x, y ∈ B *we have* F(x) ≤ cF(y) *for some*

*constant* c*. Let* X = {x1 , ··· ,xk } *be a sample of* k *i.i.d. points from* F*. Then we have*



*Proof.* Fix a value ∈ > 0 and denote the ball of radius ∈ with a center y by B∈ (y). Consider the following covering of B using balls of radius ∈ . We center the ﬁrst ball at the center of B. At the i-th iteration, if B \ ∪j<iB∈ (yj )  ∅, we pick an arbitrary point in the difference and continue the process. Clearly, this process ends in ﬁnite time as B is compact and each pair of the chosen centers have distance at least ∈ . We now prove that any ball B∈ (y) can have a non-empty in- tersection with at most 5d other balls. This is because the centers of the intersecting balls should all lie inside the ball B2∈(y). Also, any two centers have distance at least ∈ . There- fore, if we draw a ball of radius ∈/2 around all the centers of the intersecting balls, then these balls are all disjoint from each other and are all inside a bigger ball B5∈/2(y). There- fore, by a simple division of the volumes, we see that there can be at most 5d centers whose corresponding ∈-ball inter- sects with B∈ (y).

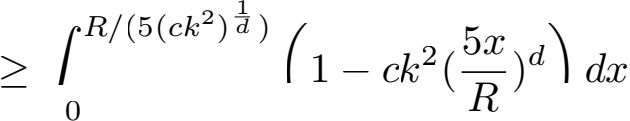
We now bound the probability that two points chosen ran- domly according F in B have distance less than ∈ . Assume that the ﬁrst chosen point is inside the ball B∈ (y). In order for the second point to be less than ∈ away from the ﬁrst one, the it should fall inside B∈ (y) or one of the intersecting balls with B∈ (y). Since we have at most 5d balls and each have measure (under F two randomly chosen balls have distance less than ∈ is upper d. By the union bound, the probability that among the k/5 i.i.d. points sampled from F at least two have distance less than ∈ is bounded upper bounded by ck2 d. As a result, denoting the minimum distance among the k/5 i.i.d. points by dmin , we obtain

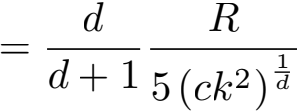


and since dmin ≥ 0 we have that

E[dmin] = ∫ 2R Pr(dmin > x)dx

0





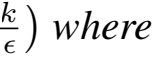
which concludes the proof ford ≥ 4. As for the cases where d = 2, 3 one can recover a similar result using a ﬁner cover-

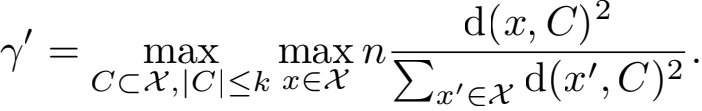
ing of the sphere.

**B Proof of Theorem 1**

**Theorem 1.** *Let* k > 0 *and* 0 < ∈ < 1*. Let* p++ (C) *be the probability of sampling a seeding* C *using* k-means++ *and* pmcmc (C) *the probability using* K-MC2 *(Algorithm 1). Then,*

Ⅱpmcmc − p++ Ⅱ TV ≤ ∈

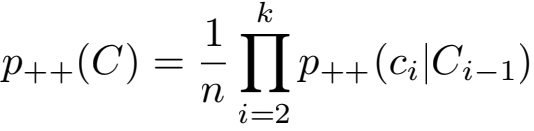
*for a chain length* m = O(γ/ log 



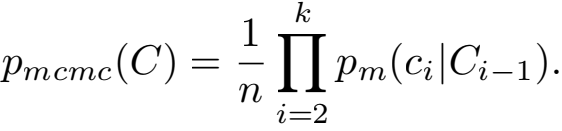
*The resulting complexity of Algorithm 1 is* O(γ/k2 d log *. Proof.* Let c1 , c2 , . . . , ck denote the k sampled cluster cen- ters in C and deﬁne for i = 1, 2, . . . , k

Ci = ∪=1 cj .

Let p++ (ci |Ci−1) denote the conditional probability of sampling ci given Ci−1 for k-means++. Similarly, pm (ci |Ci−1) denotes the conditional probability for K-MC2 with chain length m. Note that



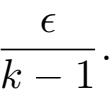
as well as



By Corollary 1 of Cai (2000) and the deﬁnition of γ/, there

exists a chain length m = O(γ/ log 

Ci−1 ⊂ X with |Ci−1| ≤ k − 1

ⅡⅡTV ≤  (8)

Next, we show an auxiliary result: Consider two arbitrary discrete probability distributions

pX,Y (x, y) = pX (x) · pY |X (y|x)

qX,Y (x, y) = qX (x) · qY |X (y|x)

with

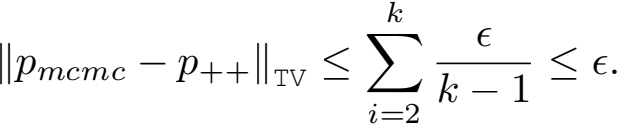
ⅡpX − qX ⅡTV ≤ ∈ 1 and **"**pX|Y − qX|Y **"**TV ≤ ∈2 .

For all x andy, it holds that

|pX,Y − qX,Y | ≤pX · |pX|Y − qX|Y | + qX|Y · |pX − qX | and we have by deﬁnition of the total variation distance

ⅡpX,Y − qX,Y ⅡTV ≤ ⅡpX − qX ⅡTV + **"**pX|Y − qX|Y **"**TV ≤ ∈ 1 + ∈2 .

An iterative application of this result to (8) yields



The same guarantee holds for the probabilities conditioned on the ﬁrst sampled center c1 , i.e.

Ⅱpmcmc (·|c1 ) − p++ (·|c1 )ⅡTV ≤ ∈ . (9)



**C Proof of Theorem 2**

**Theorem 2.** *Let* k > 0 *and* X *be a dataset with* α = O(log2 n) *and* β = O(k)*, i.e. assume* ***(A1)*** *and* ***(A2)****. Let* C *be the set of centers sampled by* K-MC2 *(Algorithm 1) with* m = O(k log2 nlog k)*. Then we have*

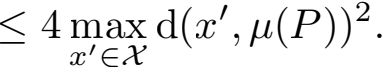
E [φC (X)] ≤ O(log k)φPT (X).

*The total complexity is* O(k3 d log2 nlog k)*.*

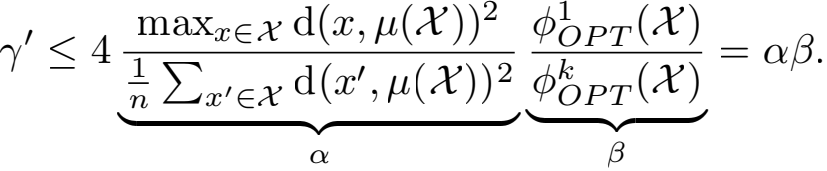
*Proof.* We have Σx∈X d(x, C)2 ≥ φPT (X) for all sets of

centers C ⊂ X of cardinality at most k. Furthermore, for all x ∈ X

d(x, C)2 ≤ 2d(x, μ(X))2 +2 d(μ(X), C)2



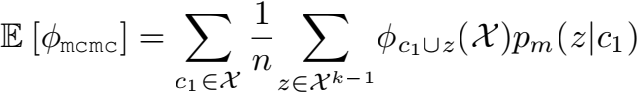
Hence,

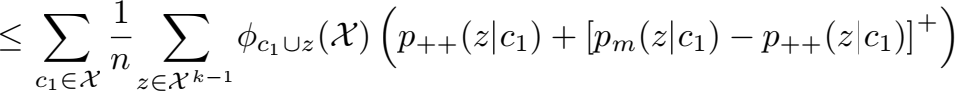


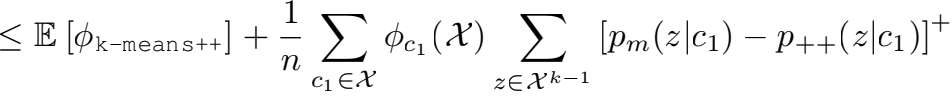
Denote by φk-means++ the quantization error for k-means++ and by φmcmc for K-MC2 . Let z be the random variable consisting of the sampled cluster cen- ters c2 , c3 , . . . , ck. Let p++ (z|c1 ) denote the conditional probability of z given the initial cluster center c1 for k-means++. Correspondingly, pm (z|c1 ) denotes the con- ditional probability for K-MC2 with chain length m. We note that pm (z|c1 ) ≤ p++ (z|c1 ) + (pm (z|c1 ) − p++ (z|c1 ))+

and E [φc1 (X)] ≤ 2βφPT (X). Using Theorem 1.1 of

Arthur and Vassilvitskii (2007) and (9), we then have that







≤ [8(log2 k +2)+2β∈/] φPT (X).

The result then follows by setting ∈/ = O(1/β).