# Tight Kernel Query Complexity of Kernel Ridge Regression and Kernel $k$-means Clustering

Manuel Fernández V
Carnegie Mellon University
manuelf@andrew.cmu.edu

David P. Woodruff
Carnegie Mellon University
dwoodruf@cs.cmu.edu

Taisuke Yasuda
Carnegie Mellon University
taisukey@andrew.cmu.edu

May 14, 2019

**Abstract**

We present tight lower bounds on the number of kernel evaluations required to approximately solve kernel ridge regression (KRR) and kernel $k$-means clustering (KKMC) on $n$ input points. For KRR, our bound for relative error approximation to the minimizer of the objective function is $\Omega(nd_{\mathrm{eff}}^{\lambda}/\varepsilon)$ where $d_{\mathrm{eff}}^{\lambda}$ is the effective statistical dimension, which is tight up to a $\log(d_{\mathrm{eff}}^{\lambda}/\varepsilon)$ factor. For KKMC, our bound for finding a $k$-clustering achieving a relative error approximation of the objective function is $\Omega(nk/\varepsilon)$, which is tight up to a $\log(k/\varepsilon)$ factor. Our KRR result resolves a variant of an open question of El Alaoui and Mahoney, asking whether the effective statistical dimension is a lower bound on the sampling complexity or not. Furthermore, for the important practical case when the input is a mixture of Gaussians, we provide a KKMC algorithm which bypasses the above lower bound.

# 1 Introduction

The *kernel trick* in machine learning is a general technique that takes linear learning algorithms that only depend on the dot products of the data, including linear regression, support vector machines, principal component analysis, and $k$-means clustering, and boosts them to powerful nonlinear algorithms. This is done by replacing the inner product between two data points with their inner product after applying a kernel map, which implicitly maps the points to a higher dimensional space via a non-linear feature map. The simplicity and power of kernel methods has lead to wide adoption across the machine learning community: nowadays, kernel methods are a staple both in theory [FHT01] and in practice [STV$^+$04, ZMLS07]. We refer the reader to [SS01] for more background on kernel methods.

However, one problem with kernel methods is that the computation of the kernel matrix $\mathbf{K}$, the matrix containing all pairs of kernel evaluations between $n$ data points, requires $\Omega(n^2)$ time, which is prohibitively expensive for the large-scale data sets encountered in modern data science. To combat this, a large body of literature in the last decade has been devoted to designing faster algorithms that attempt to trade a small amount of accuracy in exchange for speed and memory, based on techniques such as random Fourier features [RR08], sampling [Bac13, EAM15, MM17, MW17], sketching [YPW17], and incomplete Cholesky factorization [BJ02, FS01]. We refer the reader to the exposition of [MM17] for a more extensive overview of recent literature on the approximation of kernel methods.

## 1.1 Previous work on kernel query complexity

In this work, we consider lower bounds on the query complexity of the kernel matrix. The kernel query complexity is a fundamental information-theoretic parameter of kernel problems and both upper and lower bounds have been studied by a number of works [LWZ14, CBMS15, MM17, MW17].

For kernel ridge regression, a lower bound has been shown for additive error approximation of the objective function value in Corollary 8 of [CBMS15], which is a weaker approximation guarantee than what we study in this work. However, their bound is not known to be tight. Furthermore, the best known upper bounds for kernel ridge regression are in terms of a data-dependent quantity known as the *effective statistical dimension* [EAM15, MM17], on which the [CBMS15] bound does not depend. The question of whether the effective statistical dimension gives a lower bound on the sample complexity has been posed as an open question by El Alaoui and Mahoney [EAM15]. We will answer this question affirmatively under a slightly different approximation guarantee than they use, which is nevertheless satisfied by known algorithms nearly tightly, for instance by [MM17].

Another kernel problem for which lower bounds have been shown is the problem of giving a $(1 + \varepsilon)$ relative Frobenius norm error rank $k$ approximation of the kernel matrix, which has a bound of $\Omega(nk/\varepsilon)$ by Theorem 13 of [MW17]. For kernel $k$-means clustering, there are no kernel complexity lower bounds to our knowledge.

Similar cost models have also been studied in the context of semisupervised/interactive learning. Intuitively, kernel evaluations are queries that ask for the similarity between two objects, where the notion of similarity in this context is the implicit notion of similarity recognized by humans, i.e. the “crowd kernel”. In such situations, the dominant cost is the number of these queries that must be made to users, making kernel query complexity an important computational parameter. Mazumdar and Saha [MS17] study the problem of clustering under the setting where the algorithm obtains information by adaptively asking users whether two data points belong to the same cluster or not. In this setting, the dominant cost that is analyzed is the number of same-cluster queries that the algorithm must make, which exactly corresponds to the kernel query complexity of clustering a set of $n$ points drawn from $k$ distinct points with the indicator function kernel and the 0-1 loss (as opposed to $k$-means clustering, which uses the $\ell_2$ loss). In [TLB$^+$11], the authors consider the problem of learning a “crowd kernel”, where the implicit kernel function is crowdsourced and the cost is measured as the number of queries of the form “is $a$ more similar to $b$ than $c$?” rather than queries that directly access the underlying kernel evaluations.

| Kernel problem | Upper bound | Lower bound |
|---|---|---|
| KRR | $O\left(\frac{nd_{\text{eff}}^{\lambda}}{\varepsilon}\log\frac{d_{\text{eff}}^{\lambda}}{\varepsilon}\right)$ ([MM17], Theorem 15) | $\Omega\left(\frac{nd_{\text{eff}}^{\lambda}}{\varepsilon}\right)$ (this work, Theorem 3.1) |
| KKMC | $O\left(\frac{nk}{\varepsilon}\log\frac{k}{\varepsilon}\right)$ ([MM17], Theorem 16) | $\Omega\left(\frac{nk}{\varepsilon}\right)$ (this work, Theorem 4.5) |

Figure 1: Table of upper bounds and lower bounds on the kernel query complexity.

## 1.2 Our contributions

In this work, we resolve the kernel query complexity of kernel ridge regression and kernel $k$-means clustering up to $\log(d_{\text{eff}}^{\lambda}/\varepsilon)$ and $\log(k/\varepsilon)$ factors, respectively. Our lower bounds apply even to *adaptive* algorithms, that is, algorithms that are allowed to decide which kernel entries to query based on the results of previous kernel queries. This is a crucial aspect of our contributions, since some of the most efficient algorithms known for kernel ridge regression and kernel $k$-means clustering make use of adaptive queries, most notably through the use of a data-dependent sampling technique known as *ridge leverage score sampling* [MM17].

For kernel ridge regression, we present Theorem 3.1, in which we construct a distribution over kernel ridge regression instances such that any randomized algorithm requires $\Omega(nd_{\text{eff}}^{\lambda}/\varepsilon)$ adaptive kernel evaluations. This matches the upper bound given in Theorem 15 of [MM17] up to a $\log(d_{\text{eff}}^{\lambda}/\varepsilon)$ factor. Although we present the main ideas of the proof using the kernel as the dot product kernel, our proof in fact applies to any kernel that is of the form $(c_1 - c_0)\mathbb{1}(\mathbf{e}_i = \mathbf{e}_j) + c_0$ for constants $c_1 > c_0$ when restricted to the standard basis vectors (Theorem 3.7). This includes any kernel that can be written as functions of dot products and Euclidean distances, including the polynomial kernel and the Gaussian kernel. This result resolves a variant of an open question posed by [EAM15], which asks whether the effective statistical dimension is a lower bound on the sampling complexity or not. In their paper, they consider the approximation guarantee of a $(1+\varepsilon)$ relative error in the statistical risk, while we consider a $(1+\varepsilon)$ relative error approximation of the minimizer of the KRR objective function. By providing tight bounds on the query complexity in terms of the effective statistical dimension $d_{\text{eff}}^{\lambda}$, we definitively establish the fundamental importance of the quantity as a computational parameter, in addition to its established significance as a statistical parameter in the statistics literature [FHT01]. Furthermore, our result also clearly gives a lower bound on the time complexity of kernel ridge regression that matches Theorem 15 of [MM17] up to a $\tilde{O}(d_{\text{eff}}^{\lambda}/\varepsilon)$ factor for intermediate ranges of $\varepsilon$. This is in contrast to the conditional $\Omega(n^{2-o(1)})$ time complexity lower bound of [BIS17], which operates in the regime of $\varepsilon = \exp(-\omega(\log^2 n))$ for approximating the argmin of the objective function.

For kernel $k$-means clustering, we present Theorem 4.5, which shows a lower bound of $\Omega(nk/\varepsilon)$ for the problem of outputting a clustering which achieves a $(1+\varepsilon)$ relative error value in the objective function. This matches the upper bound given in Theorem 16 of [MM17] up to a $\log(k/\varepsilon)$ factor. We also note that the problem of outputting a $(1 \pm \varepsilon)$ relative error approximation of the optimal cost itself has an $O(nk) + \operatorname{poly}(k, 1/\varepsilon, \log n)$ algorithm, and we complement it with a lower bound of $\Omega(nk)$ in Proposition 4.3.

Although our lower bounds show that existing upper bounds for kernel ridge regression and kernel $k$-means clustering are optimal, up to logarithmic factors, in their query complexity, one could hope that for important input distributions that may occur in practice, that better query complexities are possible. We show specifically in the case of kernel $k$-means that when the $n$ points are drawn from a mixture of $k$ Gaussians with $1/\operatorname{poly}(k/\varepsilon)$ mixing probabilities and a separation between their means that matches the information-theoretically best possible for learning the means given by [RV17], one can bypass the $\Omega(nk/\varepsilon)$ lower bound, achieving an $(n/\varepsilon)\operatorname{poly}(\log(k/\varepsilon))$ query upper bound, effectively saving a factor of $k$ from the lower bounds for worst-case input distributions. This is our Theorem 5.1.

## 1.3 Our techniques

To prove our lower bounds, we design hard input distributions of kernel matrices as inner product matrices of an i.i.d. sample of vectors.

For our KRR lower bound, we draw our sample of vectors as follows: with probability $1/2$, we draw our vector uniformly from the first $(1/2)(k/\varepsilon)$ standard basis vectors, and with probability $1/2$, we draw our vector uniformly from the next $(1/4)(k/\varepsilon)$ standard basis vectors. Now if we draw our data set as $n$ points sampled from this distribution, then, on average, half of the input points have $n\varepsilon/k$ copies of themselves in the data set while the other half have $2n\varepsilon/k$ copies. We first show that correctly deciding between these two cases for a constant fraction of the $n$ input points requires $\Omega(nk/\varepsilon)$ queries by standard arguments. We will then show that running KRR with a regularization of $\lambda = n/k$ and a target vector of all ones can solve this problem, while having an effective statistical dimension of $\Theta(k)$, giving the desired theorem. To see this, first note that the kernel matrix $\mathbf{K}$ has rank $(3/4)(k/\varepsilon)$, where each of the $j$th nonzero eigenvalue $n_j$ is the number of copies of the $j$th standard basis vector in the data set. Then, we show that the true argmin of the KRR objective has the $i$th coordinate as $(n_j + \lambda)^{-1}$, where $n_j$ is the number of copies of the $i$th input vector. Then, if $n_j = n\varepsilon/k$, then this is $(k/n)/(1+\varepsilon)$ while if $n_j = 2n\varepsilon/k$, then this is $(k/n)/(1+2\varepsilon)$. Since these two cases are separated by a $(1 \pm \varepsilon)$ factor, a $(1+\varepsilon)$-relative approximation of the argmin can distinguish these cases for a constant fraction of coordinates by averaging.

For our KKMC lower bound, we draw our sample of vectors as follows: we first divide the coordinates of $\mathbb{R}^{k/\varepsilon}$ into $k$ blocks of size $1/\varepsilon$, uniformly select a block, uniformly select a pair of coordinates $j_1 \neq j_2$ from the block, and draw the sum of the corresponding standard basis vectors $\mathbf{e}_{j_1} + \mathbf{e}_{j_2}$. Intuitively, an optimal clustering should cluster points in the same block together, and it turns out that this clustering has a cost of $n(1-2\varepsilon)$. We first show that for any set $S$ of size at most $|S| \leq 2n/5$ points, there is a lower bound on the cost of at least $|S| - (77/40)n\varepsilon$, and that for the set $S'$ of points $\mathbf{x}$ belonging to a cluster in which uniformly sampling a point $\mathbf{x}'$ in its cluster has $\langle \mathbf{x}, \mathbf{x}' \rangle \neq 0$ with probability less than $o(\varepsilon)$, then the cost is $|S'|(1-o(\varepsilon))$. Then setting $S$ to be the complement of $S'$, i.e. points with an $\Omega(\varepsilon)$ probability of sampling a nonzero inner product, we conclude that if $|S'| \geq 3n/5$, then the cost is not within a $(1+\varepsilon/40)$ factor of the optimal cost. Thus, $|S'| \leq 3n/5$ and so for at least a $2n/5$ fraction of points, there must be an $\Omega(\varepsilon)$ probability of sampling a nonzero inner product among its cluster. However, we then show that constructing a clustering with this guarantee requires $\Omega(nk/\varepsilon)$ inner products by standard arguments, giving the theorem.

In our algorithm for mixtures of Gaussians, we exploit the input distribution itself to efficiently compute sketches $\mathbf{Sx}$ of the input points $\mathbf{x}$, where $\mathbf{S}$ is a matrix of zero mean i.i.d. Gaussians. Once we have computed these sketches, we show that we may compute an approximately optimal clustering in no more inner product evaluations.

### 1.4 Open questions

We suggest several related open questions. First, the error guarantee that we consider for the KRR lower bound does not directly measure the predictive performance of the KRR estimator. Thus, a more desirable result would be to find tight lower bounds for an algorithm outputting an estimator that guarantees, say, a $(1+\varepsilon)$ relative error guarantee for the statistical risk of the resulting estimator. This is the main error guarantee considered in [MM17] as well. Another interesting direction is to characterize the complexity of finding KRR estimators with objective function value guarantees as well, for which there are no query complexity efficient algorithms to the best of our knowledge.

A couple of other kernel problems have query complexity efficient algorithms using ridge leverage score sampling, including kernel principal component analysis and kernel canonical correlation analysis [MM17]. We leave it open to determine whether these problems have matching lower bounds as well.

### 1.5 Paper outline

In section 2, we recall basic definitions and results about KRR and KKMC that we use in our lower bound results. We then prove our KRR lower bound in section 3 and our KKMC lower bound in section 4. Finally, our query complexity efficient clustering algorithm for mixtures of Gaussians is given in section 5.

## 2 Preliminaries

### 2.1 Notation

We denote the set $\{1, 2, \dots, n\}$ by $[n]$. For $j \in [d]$, we write $\mathbf{e}_j \in \mathbb{R}^d$ for the standard Euclidean basis vectors. We write $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ for the $n \times n$ identity matrix and $\mathbf{1}_n \in \mathbb{R}^n$ for the vector of all ones in $n$ dimensions.

Let $\mathcal{S}$ be a finite set. Given two distributions $\mu, \nu$ on $\mathcal{S}$, the *total variation distance between* $\mu$ *and* $\nu$ is

$$D_{TV}(\mu, \nu) = \sum_{s \in \mathcal{S}} |\mu(s) - \nu(s)|. \tag{2.1}$$

We write $\mathrm{Unif}(\mathcal{S})$ for the uniform distribution on $\mathcal{S}$.

Let $\mathcal{X}$ be the input space of a data set and $\mathcal{F}$ a reproducing kernel Hilbert space with kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. We write $\varphi : \mathcal{X} \to \mathcal{F}$ for the feature map, i.e. the $\varphi$ such that $k(\mathbf{x}, \mathbf{y}) = \langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle_{\mathcal{F}}$. For a set of vectors $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X}$ and a kernel map $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we write $\mathbf{K} \in \mathbb{R}^{n \times n}$ for the kernel matrix, i.e. the matrix with $\mathbf{e}_i^\top \mathbf{K} \mathbf{e}_j \coloneqq k(\mathbf{x}_i, \mathbf{x}_j)$. Note that $\mathbf{K}$ is symmetric and positive semidefinite (PSD). We refer the reader to [SS01] for more details on the general theory of kernel methods. For all of our lower bound constructions, we will take $\mathcal{X} = \mathbb{R}^d$ and our kernel to be the linear kernel, i.e. the standard dot product on $\mathbb{R}^d$, $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$. Hence, we will frequently refer to kernel queries alternatively as inner product queries.

## 2.2 Kernel ridge regression

The kernel ridge regression (KRR) task is defined as follows. We parameterize an instance of KRR by a triple $(\mathbf{K}, \mathbf{z}, \lambda)$, where $\mathbf{K} \in \mathbb{R}^n$ is the kernel matrix of a data set $\{\mathbf{x}_i\}_{i=1}^n$, $\mathbf{z} \in \mathbb{R}^n$ is the target vector, and $\lambda$ is the regularization parameter. The problem is to compute

$$\boldsymbol{\alpha}_{\mathrm{opt}} \coloneqq \operatorname*{argmin}_{\boldsymbol{\alpha} \in \mathbb{R}^n} \|\mathbf{K}\boldsymbol{\alpha} - \mathbf{z}\|_2^2 + \lambda \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}. \tag{2.2}$$

It is well-known that the solution to the above is given in closed form by

$$\boldsymbol{\alpha}_{\mathrm{opt}} = (\mathbf{K} + \lambda \mathbf{I}_n)^{-1} \mathbf{z} \tag{2.3}$$

which can be shown for example by completing the square.

An important parameter to the KRR instance $(\mathbf{K}, \mathbf{z}, \lambda)$ is the *effective statistical dimension*:

**Definition 2.1** (Effective statistical dimension ([FHT01, Zha05]). *Given a rank $r$ kernel matrix $\mathbf{K}$ with eigenvalues $\sigma_i^2$ for $i \in [r]$ and a regularization parameter $\lambda$, we define the effective statistical dimension as*

$$d_{\mathrm{eff}}^{\lambda}(\mathbf{K}) \coloneqq \operatorname{tr}\Big(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I}_n)^{-1}\Big) = \sum_{i=1}^{r} \frac{\sigma_i^2}{\sigma_i^2 + \lambda}. \tag{2.4}$$

*We simply write $d_{\mathrm{eff}}^{\lambda}$ when the kernel matrix $\mathbf{K}$ is clear from context.*

The effective statistical dimension was first introduced to measure the statistical capacity of the KRR instance, but has since been used to parameterize its computational properties as well, in the form of bounds on sketching dimension [ACW17] and sampling complexity [EAM15, MM17].

### 2.2.1 Approximate solutions

In the literature, various notions of approximation guarantees for KRR have been studied, including $(1+\varepsilon)$ relative error approximations in the objective function cost [ACW17] and $(1+\varepsilon)$ relative error approximations in the statistical risk [Bac13, EAM15, MM17]. In our paper, we consider a slightly different approximation guarantee, namely a $(1+\varepsilon)$ relative error approximation of the argmin of the KRR objective function.

**Definition 2.2** ($(1+\varepsilon)$-approximate solution to kernel ridge regression). *Given a kernel ridge regression instance $(\mathbf{K}, \mathbf{z}, \lambda)$, we say that $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ is a $(1+\varepsilon)$-approximate solution to kernel ridge regression if*

$$\|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}_{\mathrm{opt}}\|_2 \le \varepsilon \|\boldsymbol{\alpha}_{\mathrm{opt}}\|_2 = \varepsilon \big\|(\mathbf{K} + \lambda \mathbf{I}_n)^{-1} \mathbf{z}\big\|_2. \tag{2.5}$$

This approximation guarantee is natural, and we note that it is achieved by the estimator of [MM17]. This guarantee is then used to prove their main statistical risk guarantee. We will briefly reproduce the proof of this fact from Theorem 15 of [MM17] below for completeness. Indeed, using a spectral approximation $\tilde{\mathbf{K}}$ satisfying $\mathbf{K} - \tilde{\mathbf{K}} \preceq \lambda \varepsilon \mathbf{I}_n$, they output an estimator $\hat{\boldsymbol{\alpha}} \coloneqq (\tilde{\mathbf{K}} + \lambda \mathbf{I}_n)^{-1} \mathbf{z}$ which satisfies the guarantee of equation (2.5) since

$$\begin{aligned}
\|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}_{\text{opt}}\|_2 &= \left\|(\tilde{\mathbf{K}} + \lambda\mathbf{I}_n)^{-1}\mathbf{z} - (\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{z}\right\|_2 \\
&= \left\|(\tilde{\mathbf{K}} + \lambda\mathbf{I}_n)^{-1}[(\mathbf{K} + \lambda\mathbf{I}_n) - (\tilde{\mathbf{K}} + \lambda\mathbf{I}_n)](\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{z}\right\|_2 \\
&= \left\|(\tilde{\mathbf{K}} + \lambda\mathbf{I}_n)^{-1}[\mathbf{K} - \tilde{\mathbf{K}}](\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{z}\right\|_2 \\
&\leq \left\|(\tilde{\mathbf{K}} + \lambda\mathbf{I}_n)^{-1}\right\|_2 \left\|\mathbf{K} - \tilde{\mathbf{K}}\right\|_2 \left\|(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{z}\right\|_2 \\
&\leq \frac{1}{\lambda}(\lambda\varepsilon)\|\boldsymbol{\alpha}_{\text{opt}}\|_2 = \varepsilon\|\boldsymbol{\alpha}_{\text{opt}}\|_2.
\end{aligned} \tag{2.6}$$

## 2.3 Kernel $k$-means clustering

Recall the feature map $\varphi : \mathcal{X} \to \mathcal{F}$ for an input space $\mathcal{X}$ and a reproducing kernel Hilbert space $\mathcal{F}$. The problem of kernel $k$-means clustering (KKMC) involves forming a partition of the data set $\{\mathbf{x}_i\}_{i=1}^n$ into $k$ clusters $\mathcal{C} \coloneqq \{C_j\}_{j=1}^k$ with centroids $\boldsymbol{\mu}_j \coloneqq \frac{1}{|C_j|}\sum_{\mathbf{x}\in C_j}\varphi(\mathbf{x})$ such that the objective function

$$\text{cost}(\mathcal{C}) \coloneqq \sum_{j=1}^k \sum_{\mathbf{x}\in C_j} \left\|\varphi(\mathbf{x}) - \boldsymbol{\mu}_j\right\|_{\mathcal{F}}^2 \tag{2.7}$$

is minimized. The problem of finding exact solutions are known to be NP-hard [ADHP09], but it has nonetheless proven to be an extremely popular model in practice [Har75].

With an abuse of notation, we will also talk about the cost of a single cluster, which is just the above sum taken only over one cluster:

$$\text{cost}(C_j) \coloneqq \sum_{\mathbf{x}\in C_j} \left\|\varphi(\mathbf{x}) - \boldsymbol{\mu}_j\right\|_{\mathcal{F}}^2. \tag{2.8}$$

As done in [BDM09, CEM$^+$15, MM17] and many other works, we consider the approximation guarantee of finding a clustering that achieves a $(1+\varepsilon)$ relative error in the objective function cost, i.e. a finding a partition $\{C'_j\}_{j=1}^k$ such that

$$\begin{aligned}
\text{cost}(\{C'_j\}_{j=1}^k) &= \sum_{j=1}^k \sum_{\mathbf{x}\in C'_j} \left\|\varphi(\mathbf{x}) - \boldsymbol{\mu}_j\right\|_{\mathcal{F}}^2 \\
&\leq (1+\varepsilon)\min_{\mathcal{C}} \text{cost}(\mathcal{C}) = (1+\varepsilon)\min_{\mathcal{C}=\{C_j\}_{j=1}^k} \sum_{j=1}^k \sum_{\mathbf{x}\in C_j} \left\|\varphi(\mathbf{x}) - \boldsymbol{\mu}_j\right\|_{\mathcal{F}}^2.
\end{aligned} \tag{2.9}$$

# 3 Lower bound for kernel ridge regression

We present our lower bound on the number of kernel entries required in order to compute a $(1+\varepsilon)$-approximate solution to kernel ridge regression (see definition 2.2).

**Theorem 3.1** (Query lower bound for kernel ridge regression)**.** *Consider a possibly randomized algorithm $\mathcal{A}$ that correctly outputs a $(1+\varepsilon)$-approximate solution $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ (see definition 2.2) to any kernel ridge regression instance $(\mathbf{K}, \mathbf{z}, \lambda)$ with probability at least $2/3$. Then there exists an input instance $(\mathbf{K}, \mathbf{z}, \lambda)$ on which $\mathcal{A}$ reads at least $\Omega(nd_{\text{eff}}^\lambda/\varepsilon)$ entries of $\mathbf{K}$, possibly adaptively, in expectation.*

Our lower bound is nearly optimal, matching the ridge leverage score algorithm in Theorem 15 of [MM17] which reads $O\Big(\frac{nd_{\text{eff}}^\lambda}{\varepsilon}\log\frac{nd_{\text{eff}}^\lambda}{\varepsilon}\Big)$ kernel entries up to a $\log\frac{d_{\text{eff}}^\lambda}{\varepsilon}$ factor.

## 3.1 Main lower bound

**Definition 3.2** (Hard input distribution – kernel ridge regression)**.** *Let $J, n \in \mathbb{N}$ and assume for simplicity that $4 \mid J$. We define a distribution $\mu_{\mathrm{KRR}}(n, J)$ on binary PSD matrices $\mathbf{K} \in \mathbb{R}^{n\times n}$ defined as follows. We first define a distribution $\nu_{\mathrm{KRR}}(J)$ over standard basis vectors $\{\mathbf{e}_j \in \mathbb{R}^{3J/4} : j \in [3J/4]\}$, where with probability $1/2$ we draw a uniformly random $\mathbf{e}_j$ from $S_1 := \{\mathbf{e}_j : j \in [J/2]\}$ and with probability $1/2$ we draw a uniformly random $\mathbf{e}_j$ from $S_2 := \{\mathbf{e}_{j+J/2} : j \in [J/4]\}$. We then generate $\mathbf{K}$ by drawing $n$ i.i.d. samples $\{\mathbf{x}_i\}_{i=1}^n$ from $\nu_{\mathrm{KRR}}(J)$ and letting $\mathbf{K}$ be the inner product matrix of $\{\mathbf{x}_i\}_{i=1}^n$, that is, $\mathbf{e}_i^\top \mathbf{K} \mathbf{e}_j := \mathbf{x}_i \cdot \mathbf{x}_j$.*

**Theorem 3.3.** *Let $\varepsilon \in (0, 1/2)$ and $J = k/\varepsilon$ with $J^2 = O(n)$ and $k$ a parameter. Suppose that there exists a possibly randomized algorithm $\mathcal{A}$ that, with probability at least $2/3$ over its random coin tosses and random kernel matrix drawn from $\mathbf{K} \sim \mu_{\mathrm{KRR}}(n, J)$, correctly outputs a $(1 + \varepsilon/100)$-approximate solution $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ (see definition 2.2) to the kernel ridge regression instance $(\mathbf{K}, \mathbf{z}, \lambda)$ with $\mathbf{z} = \mathbf{1}_n$ and $\lambda = n/k$. Furthermore, suppose that $\mathcal{A}$ reads at most $r$ positions of $\mathbf{K}$ on any input, possibly adaptively. Then, $d_{\mathrm{eff}}^{\lambda}(\mathbf{K}) = \Theta(k)$ and $r = \Omega(n d_{\mathrm{eff}}^{\lambda}/\varepsilon)$.*

To prove Theorem 3.3, we will make a reduction to the following hardness lemma.

**Lemma 3.4.** *Recall the definitions of $\mu_{\mathrm{KRR}}(n, J), \nu_{\mathrm{KRR}}(J), S_1, S_2$ from definition 3.2. Suppose that there exists a possibly randomized algorithm $\mathcal{A}$ that, with probability at least $2/3$ over its random coin tosses and random inputs drawn from $\mu_{\mathrm{KRR}}(n, k/\varepsilon)$, correctly outputs whether $\mathbf{x}_i$ corresponds to $\mathbf{e}_j$ with $j \in S_1$ or $j \in S_2$ for at least a $9/10$ fraction of rows $\mathbf{e}_i^\top \mathbf{K}$ for $i \in [n]$. Further, suppose that $\mathcal{A}$ reads at most $r$ positions of $\mathbf{K}$ on any input, possibly adaptively. Then, $r = \Omega(nJ)$.*

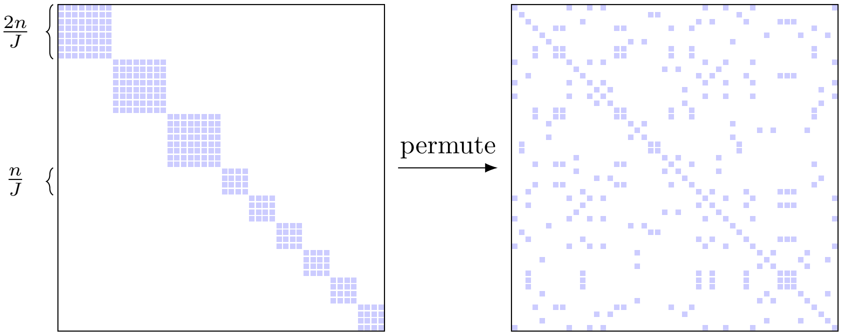

Figure 2: The hardness lemma – does the $i$th row have $\frac{2n}{J}$ or $\frac{n}{J}$ ones?

*Proof of Lemma 3.4.* First consider a single draw $\mathbf{x} \sim \nu_{\mathrm{KRR}}(J)$. We claim that $\Omega(J)$ adaptive inner product queries $\mathbf{x} \cdot \mathbf{e}_j$ are required to correctly output whether $\mathbf{x} \in S_1$ or $\mathbf{x} \in S_2$ with probability at least $2/3$ over $\nu_{\mathrm{KRR}}(J)$. Suppose there exists a randomized algorithm $\mathcal{B}$ that has the above guarantee. By Yao's minimax principle [Yao77], there then exists a deterministic algorithm $\mathcal{B}'$ with the same guarantee and the same expected cost over the input distribution. Then, $\mathcal{B}'$ can be used to construct a hypothesis test to decide whether $\mathbf{x} \sim \mathrm{Unif}(S_1)$ or $\mathbf{x} \sim \mathrm{Unif}(S_2)$ which succeeds with probability at least $2/3$. Now let $S$ denote the random variable indicating the list of inner product queries made and their corresponding values, let $L_1$ denote the distribution of $S$ conditioned on $\mathbf{x} \sim \mathrm{Unif}(S_1)$, and $L_2$ the distribution of $S$ conditioned on $\mathbf{x} \sim \mathrm{Unif}(S_2)$. Then by Proposition 2.58 of [BYP02], we have that

$$\frac{1 + D_{TV}(L_1, L_2)}{2} \geq \frac{2}{3} \tag{3.1}$$

and thus rearranging gives $D_{TV}(L_1, L_2) \geq 1/3$. Now suppose for contradiction that $\mathcal{B}'$ makes at most $q \leq J/100$ queries on any input. Since $\mathcal{B}'$ is deterministic, it makes the same sequence of inner product queries $\mathbf{x} \cdot \mathbf{e}_{j_1}, \mathbf{x} \cdot \mathbf{e}_{j_2}, \ldots, \mathbf{x} \cdot \mathbf{e}_{j_q}$, if it reads a sequence of $q$ zeros. Now fix these queries $j_1, j_2, \ldots, j_q$. We then have that for each $\ell \in [q]$,

$$\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(\mathbf{x} = \mathbf{e}_{j_\ell}) = \frac{1}{J}, \qquad \mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(\mathbf{x} = \mathbf{e}_{j_\ell}) = \frac{2}{J} \tag{3.2}$$

and thus by the union bound,

$$\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(\mathbf{x}\in\{\mathbf{e}_{j_\ell}:\ell\in[q]\})\le\frac{q}{J},\qquad\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(\mathbf{x}\in\{\mathbf{e}_{j_\ell}:\ell\in[q]\})\le\frac{2q}{J}.\tag{3.3}$$

Now let $\Omega$ denote the support of $S$ and let $s_0\in\Omega$ denote the value of $S$ when $\mathcal{B}'$ reads all zeros. Then,

$$\begin{aligned}D_{TV}(L_1,L_2)&=\sum_{s\in\Omega}\left|\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(S=s)-\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(S=s)\right|\\&=\left|\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(S=s_0)-\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(S=s_0)\right|+\sum_{s\in\Omega\setminus\{s_0\}}\left|\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(S=s)-\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(S=s)\right|\\&\le\frac{2q}{J}+\sum_{s\in\Omega\setminus\{s_0\}}\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(S=s)+\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(S=s)\\&=\frac{2q}{J}+\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_1)}(S\neq s_0)+\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\mathrm{Unif}(S_2)}(S\neq s_0)\le\frac{2q}{J}+\frac{q}{J}+\frac{2q}{J}=\frac{5q}{J}\le\frac{1}{20}\end{aligned}\tag{3.4}$$

which contradicts $D_{TV}(L_1,L_2)\ge 1/3$. Thus, we conclude that $q>J/100$.

We now prove the full claim via a reduction to the above problem of deciding whether some $\mathbf{x}\sim\nu_{\mathrm{KRR}}(J)$ is either drawn from $S_1$ or $S_2$. Suppose for contradiction that there exists a randomized algorithm $\mathcal{A}$ with the guarantees of the lemma which reads $r=o(nJ)$. We then design an algorithm $\mathcal{B}$ using $\mathcal{A}$ as follows. We independently sample a uniformly random index $i^*\sim\mathrm{Unif}([n])$ and $n-1$ points $\{\mathbf{x}_i\}_{i=1}^{n-1}$ with $\mathbf{x}_i\sim\nu_{\mathrm{KRR}}(J)$ for each $i\in[n-1]$. We then run $\mathcal{A}$ on the kernel matrix instance $\mathbf{K}$ corresponding to setting the $i^*$th standard basis vector to $\mathbf{x}$ and the other $n-1$ vectors according to $\{\mathbf{x}_i\}_{i=1}^{n-1}$. Note then that we can generate any entry of $\mathbf{K}$ on row $i^*$ or column $i^*$ by an inner product query to $\mathbf{x}$, and otherwise we can simulate the kernel query without making any inner product queries to $\mathbf{x}$. If $\mathcal{A}$ ever reads more than $J/100$ entries of $\mathbf{x}$, we output failure. Since $r=o(nJ)$, by averaging, for at least a $199/200$ fraction of the $n$ rows of $\mathbf{K}$, $\mathcal{A}$ reads at most $J/200$ entries of the row $\mathbf{e}_i^\top\mathbf{K}$. Similarly, for at least a $199/200$ fraction of the $n$ columns of $\mathbf{K}$, $\mathcal{A}$ reads at most $J/200$ entries of the column $\mathbf{K}\mathbf{e}_i$. Thus, for at least a $99/100$ fraction of the input points, $\mathcal{A}$ makes at most $J/100$ inner product queries. It follows by symmetry that with probability $99/100$, $\mathcal{A}$ makes at most $J/100$ inner product queries on $\mathbf{x}$. Then by a union bound over the random choice of $i^*$ over the $n$ input points, $\mathcal{A}$ correctly decides whether $\mathbf{x}\sim\mathrm{Unif}(S_1)$ or $\mathbf{x}\sim\mathrm{Unif}(S_2)$ and attempts to read at most $J/100$ entries of $\mathbf{x}$ with probability at least $1/10+1/100=11/100$. Thus, $\mathcal{B}$ succeeds with probability at least $1-11/100\ge 2/3$, contradicting the above result. □

With Lemma 3.4 in hand, we finally get to the proof of Theorem 3.3.

*Proof of Theorem 3.3.* Assume that $nJ=o(n^2)$, since otherwise the lower bound is $\Omega(n^2)$, which is best possible. Note that for $\mathbf{x}\sim\nu_{\mathrm{KRR}}(J)$, $\mathbf{x}=\mathbf{e}_j$ with probability $\frac{1}{2}\frac{1}{J/2}=\frac{1}{J}$ if $\mathbf{e}_j\in S_1$ and $\frac{1}{2}\frac{1}{J/4}=\frac{2}{J}$ if $\mathbf{e}_j\in S_2$. For a fixed $j\in[3J/4]$, let $n_j$ be the number of $\mathbf{e}_j$ sampled in $\mathbf{K}$ and $\mu_j:=\mathbf{E}_{\mathbf{K}\sim\mu_{\mathrm{KRR}}(n,J)}(n_j)$. Note that $\mu_j=n/J$ for $j\in[J/2]$ and $\mu_j=2n/J$ for $j\in[J/4]+J/2$. Then by Chernoff bounds,

$$\mathop{\mathbf{Pr}}_{\mathbf{K}\sim\mu_{\mathrm{KRR}}(n,J)}\left(\left\{|n_j-\mu_j|\ge\frac{1}{100}\mu_j\right\}\right)\le 2\exp\left(-\frac{1}{100}\frac{\mu_j}{3}\right)\le 2\exp\left(-\frac{1}{100}\frac{n/J}{3}\right)\tag{3.5}$$

so by a union bound, we have that

$$\mathop{\mathbf{Pr}}_{\mathbf{K}\sim\mu_{\mathrm{KRR}}(n,J)}\left(\bigcup_{j\in[3J/4]}\left\{|n_j-\mu_j|\ge\frac{1}{100}\mu_j\right\}\right)\le 2\frac{3J}{4}\exp\left(-\frac{1}{100}\frac{n/J}{3}\right).\tag{3.6}$$

Since $nJ=o(n^2)$, we have that $n/J=\omega(1)$. Furthermore, since $J^2=O(n)$, we have that $J=O(n/J)$. Thus, the above happens with probability at most $1/100$ by taking $n/J$ large enough. Dismiss this event as a failure and assume that $|n_j-\mu_j|\le\frac{1}{100}\mu_j$ for all $j\in[3J/4]$.

Now let $\mathbf{K} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^\top$ be the full SVD of $\mathbf{K}$. Note that the first $3J/4$ singular values are $n_j$ with corresponding singular vectors $\mathbf{U}\mathbf{e}_j = \frac{1}{\sqrt{n_j}}\mathbf{K}\mathbf{e}_j$, and the rest are all 0s. Then, the target vector $\mathbf{z} = \mathbf{1}_n$ can be written as

$$\mathbf{z} = \sum_{j\in[3J/4]} \mathbf{K}\mathbf{e}_j = \sum_{j\in[3J/4]} \sqrt{n_j}\mathbf{U}\mathbf{e}_j, \tag{3.7}$$

since each coordinate $i \in [n]$ belongs to exactly one of the $3J/4$ input points drawn from $\nu_{\mathrm{KRR}}(n, J)$. The optimal solution can then be written as

$$\begin{aligned}\boldsymbol{\alpha}_{\mathrm{opt}} &= (\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{z} = \mathbf{U}(\boldsymbol{\Sigma} + \lambda\mathbf{I}_n)^{-1}\mathbf{U}^\top\mathbf{z} \\ &= \sum_{j\in[3J/4]} \sqrt{n_j}\mathbf{U}(\boldsymbol{\Sigma} + \lambda\mathbf{I}_n)^{-1}\mathbf{U}^\top\mathbf{U}\mathbf{e}_j = \sum_{j\in[3J/4]} \frac{1}{n_j+\lambda}\bigl(\sqrt{n_j}\mathbf{U}\mathbf{e}_j\bigr).\end{aligned} \tag{3.8}$$

Thus, for $i \in [n]$, the optimal solution takes the value $(\boldsymbol{\alpha}_{\mathrm{opt}})_i = (n_{j_i} + \lambda)^{-1}$ where $j_i \in [3J/4]$ is the index of the standard basis vector that the $i$th input point corresponds to.

Now by multiplying the $(1 + \varepsilon/100)$-approximation guarantee by $n/k$ and squaring, we have that

$$\left\|\frac{n}{k}\hat{\boldsymbol{\alpha}} - \frac{n}{k}\boldsymbol{\alpha}_{\mathrm{opt}}\right\|_2^2 \le \frac{\varepsilon^2}{100^2}\left\|\frac{n}{k}\boldsymbol{\alpha}_{\mathrm{opt}}\right\|_2^2 = \frac{\varepsilon^2}{100^2}\sum_{j\in[3J/4]}\left\|\frac{n/k}{n_j+\lambda}\bigl(\sqrt{n_j}\mathbf{U}\mathbf{e}_j\bigr)\right\|_2^2 \le \frac{\varepsilon^2}{100^2}\|\mathbf{z}\|_2^2 = \frac{\varepsilon^2}{100^2}n \tag{3.9}$$

so by averaging, we have that $\bigl(\frac{n}{k}(\hat{\boldsymbol{\alpha}})_i - \frac{n}{k}(\boldsymbol{\alpha}_{\mathrm{opt}})_i\bigr)^2 \le \varepsilon^2/100$ for at least a 99/100 fraction of the $n$ coordinates of $i$. Then on these coordinates, $\bigl|\frac{n}{k}(\hat{\boldsymbol{\alpha}})_i - \frac{n}{k}(\boldsymbol{\alpha}_{\mathrm{opt}})_i\bigr| \le \varepsilon/10$. Now note that on these coordinates, we have that

$$\begin{aligned}\left|\frac{n}{k}(\hat{\boldsymbol{\alpha}})_i - \frac{n}{k}\frac{1}{\mu_j+\lambda}\right| &\le \left|\frac{n}{k}(\hat{\boldsymbol{\alpha}})_i - \frac{n}{k}(\boldsymbol{\alpha}_{\mathrm{opt}})_i\right| + \left|\frac{n}{k}(\boldsymbol{\alpha}_{\mathrm{opt}})_i - \frac{n}{k}\frac{1}{\mu_j+\lambda}\right| \\ &\le \frac{\varepsilon}{10} + \frac{n}{k}\left|\frac{1}{n_j+n/k} - \frac{1}{\mu_j+n/k}\right| \le \frac{\varepsilon}{10} + \frac{n}{k}\frac{|n_j-\mu_j|}{(n_j+n/k)(\mu_j+n/k)} \\ &\le \frac{\varepsilon}{10} + \frac{\mu_j/100}{n/k} \le \frac{\varepsilon}{10} + \frac{2n\varepsilon/(100k)}{n/k} = \frac{6}{50}\varepsilon.\end{aligned} \tag{3.10}$$

Since

$$\frac{n}{k}\frac{1}{n\varepsilon/k+n/k} - \frac{n}{k}\frac{1}{2n\varepsilon/k+n/k} = \frac{1}{1+\varepsilon} - \frac{1}{1+2\varepsilon} = \frac{\varepsilon}{(1+\varepsilon)(1+2\varepsilon)} > \frac{\varepsilon}{3} > 2\frac{6}{50}\varepsilon \tag{3.11}$$

for $\varepsilon \in (0, 1/2)$, we can distinguish whether the $i$th input point has $\mu_j = n\varepsilon/k$ or $\mu_j = 2n\varepsilon/k$ on these coordinates and thus we can solve the hard computational problem of Lemma 3.4 without reading anymore entries of $\mathbf{K}$ after solving the kernel ridge regression instance. Thus, we have that $\mathcal{A}$ reads $\Omega(nk/\varepsilon)$ kernel entries by a reduction to Lemma 3.4.

Finally, to obtain the statement of the theorem, it remains to show that $d_{\mathrm{eff}}^\lambda = \Theta(k)$. Indeed,

$$d_{\mathrm{eff}}^\lambda = \sum_{j\in[3J/4]}\frac{n_j}{n_j+\lambda} = \Theta\left(\sum_{j\in[3J/4]}\frac{n\varepsilon/k}{n\varepsilon/k+n/k}\right) = \Theta(k) \tag{3.12}$$

as desired. □

We now obtain Theorem 3.1 by scaling parameters by constant factors.

**Remark 3.5.** *The setting of the regularization parameter in the above construction is a bit unnatural as the top $d_{\mathrm{eff}}^\lambda = \Theta(k)$ singular values of the kernel matrix are of order $n\varepsilon/k$ while the regularization is of order $n/k$, which is $1/\varepsilon$ times larger. One can easily fix this as follows. We add $(n/k)\mathbf{e}_i$ to the end of our data set for $i = k/\varepsilon + 1, k/\varepsilon + 2, \ldots, k/\varepsilon + k$. This only increases our effective statistical dimension to*

$$d_{\mathrm{eff}}^\lambda = \sum_{j\in[3J/4]}\frac{n_j}{n_j+\lambda} + \sum_{i=1}^{k}\frac{n/k}{n/k+\lambda} = \Theta\left(\sum_{j\in[3J/4]}\frac{n\varepsilon/k}{n\varepsilon/k+n/k} + \frac{k}{2}\right) = \Theta(k) \tag{3.13}$$

*and our hardness argument is clearly unaffected. Now the setting of the regularization is such that it scales as the top $d_{\mathrm{eff}}^\lambda$ singular values, so that it reduces the effects of the next $k/\varepsilon$ noisy directions, which is natural.*

## 3.2 Extensions to other kernels

The above lower bound was proven just for the dot product kernel, but we note that essentially the same proof applies to more general kernels as well. To this end, we introduce the notion of *indicator kernels*:

**Definition 3.6** (Indicator kernels)**.** *We say that* $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ *is an* indicator kernel *if there exist* $c_1 > 0$ *and* $c_0 < c_1$ *such that*

$$k(\mathbf{e}_i, \mathbf{e}_j) = \begin{cases} c_1 & \textit{if } i = j \\ c_0 & \textit{otherwise} \end{cases} \tag{3.14}$$

*for all standard basis vectors* $\mathbf{e}_i, \mathbf{e}_j$ *for* $i, j \in [d]$.

Examples of such kernels include generalized dot product kernels and distance kernels, i.e. kernels of the form $k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x} \cdot \mathbf{x}')$ and $k(\mathbf{x}, \mathbf{x}') = f(\|\mathbf{x} - \mathbf{x}'\|_2)$ for an appropriate function $f : \mathbb{R} \to \mathbb{R}$, which in turn include important kernels such as the polynomial kernel, the Gaussian kernel, etc.

Note that if $c_0 = 0$, the kernel matrix is just $c_1$ times the kernel matrix from before, so it is easy to see that the exact same proof works after scaling $\lambda$ by $c_1$. When $c_0$ is nonzero, then every entry of the kernel matrix is offset by $c_0$. However we will see that even in this case, the same proof still applies.

**Theorem 3.7** (Query lower bound for kernel ridge regression for indicator kernels)**.** *The lower bound of Theorem 3.1 continues to hold for any algorithm computing a* $(1+\varepsilon)$ *relative error solution to a KRR instance with an indicator kernel (Definition 3.6) instead of the dot product kernel.*

*Proof.* Suppose we draw our kernel $\mathbf{K}$ as in Definition 3.2, with the dot product kernel being replaced by any indicator kernel. Let $\mathbf{G}$ be the inner product matrix of the point set with SVD $\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$ as before. Then, we may write the kernel matrix as

$$\mathbf{K} = c_0 \mathbf{1}_{n\times n} + (c_1 - c_0)\mathbf{G}. \tag{3.15}$$

Now define

$$C \coloneqq c_0 \mathbf{1}_n ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1} \mathbf{1}_\mathbf{n}. \tag{3.16}$$

Then, by the Sherman-Morrison formula, $C \neq -1$ since $(\mathbf{K} + \lambda \mathbf{I}_n)$ is invertible, and so we have that

$$\begin{aligned}
\boldsymbol{\alpha}_{\mathrm{opt}} &= (\mathbf{K} + \lambda \mathbf{I}_n)^{-1}\mathbf{z} \\
&= (c_0 \mathbf{1}_n \mathbf{1}_n^\top + (c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{z} \\
&= ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{z} - \frac{((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}(c_0 \mathbf{1}_n \mathbf{1}_n^\top)((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}}{1 + c_0 \mathbf{1}_n^\top ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{1}_n} \mathbf{1}_n \\
&= ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{z} - ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{1}_n \frac{c_0 \mathbf{1}_n^\top ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{1}_n}{1 + c_0 \mathbf{1}_n^\top ((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{1}_n} \\
&= \left(1 - \frac{C}{1 + C}\right)((c_1 - c_0)\mathbf{G} + \lambda \mathbf{I}_n)^{-1}\mathbf{z} \\
&= \frac{1}{(c_1 - c_0)(1 + C)}(\mathbf{G} + (\lambda/(c_1 - c_0))\mathbf{I}_n)^{-1}\mathbf{z}
\end{aligned} \tag{3.17}$$

Thus, we find that the exact same proof as before works by setting $\lambda = (c_1 - c_0)n/k$. □

# 4 Lower bound for kernel $k$-means clustering

## 4.1 Finding the cost vs. assigning points

Recall that [MM17] present an algorithm for solving KKMC with a kernel querying complexity of $O\big(\frac{nk}{\varepsilon} \log \frac{k}{\varepsilon}\big)$. We now briefly present some intuition on how we would like to match this up to $\log \frac{k}{\varepsilon}$. We first note that the hardness cannot come from finding the centers of an approximately optimal clustering or approximating the cost of the optimal clustering up to $(1 \pm \varepsilon)$, since there is an algorithm for finding these in $O(nk +$

$\mathrm{poly}(k, 1/\varepsilon, \log n))$ kernel queries: indeed, Theorem 15.5 of [FL11] shows how to find a strong $\varepsilon$-coreset of size $\mathrm{poly}(k \log n/\varepsilon)$ in $O(nk + \mathrm{poly}(k, 1/\varepsilon, \log n))$ kernel queries, which can then be used to compute both approximate centers and the cost. Thus, intuitively, in order to achieve a lower bound of $\Omega(nk/\varepsilon)$ which nearly matches the dominant term in the upper bound of [MM17], we must design a hard point set in which the hardness is not in computing the cost nor the centers, but rather in assigning the $n$ input points to their appropriate clusters.

We take this opportunity to prove a lower bound of $\Omega(nk)$ kernel queries for the problem of computing a $(1+\varepsilon)$ relative error approximation to the cost of KKMC. In practical applications, the $nk$ term dominates the $\mathrm{poly}(k, 1/\varepsilon, \log n)$ term and thus we obtain a fairly tight characterization of this subproblem of KKMC. To prove this result, we make use of the hardness of deciding whether a binary PSD matrix has rank $k$ or $k+1$.

**Definition 4.1** (Hard input distribution – rank). *Consider the distribution $\mu_{\mathrm{rank}}(n,k)$ on binary PSD matrices $\mathbf{K} \in \mathbb{R}^{n\times n}$ defined as follows. We first draw $n$ i.i.d. samples $\{\mathbf{x}_i\}_{i=1}^n$ drawn from $\mathrm{Unif}(\{\mathbf{e}_j : j \in [k]\})$. Then, with probability $1/2$, select a uniformly random index $i^* \in [n]$ and set $\mathbf{x}_i \coloneqq \mathbf{e}_{k+1}$. Finally, generate $\mathbf{K}$ as the inner product matrix of $\{\mathbf{x}_i\}_{i=1}^n$, that is, $\mathbf{e}_i^\top \mathbf{K} \mathbf{e}_j \coloneqq \mathbf{x}_i \cdot \mathbf{x}_j$.*

**Lemma 4.2.** *Suppose there exists an algorithm which, with probability at least $5/8$, over its random coin tosses and random inner product matrix $\mathbf{K} \sim \mu_{\mathrm{rank}}(n,k)$, correctly decides whether $\mathbf{K}$ has rank $k+1$ or at most $k$. Furthermore, suppose that the algorithm reads at most $r$ positions of $\mathbf{K}$, possibly adaptively. Then, $r = \Omega(nk)$.*

*Proof.* Suppose for contradiction that $r = o(nk)$. Let $\mathcal{S}_k \coloneqq \{\mathbf{e}_j : j \in [k]\}$. We consider the following hypothesis test: decide whether some $\mathbf{x}$ is drawn from $\mathbf{x} \sim \mathrm{Unif}(\mathcal{S}_k)$ or from $\mathbf{x} = \mathbf{e}_{k+1}$ using inner product queries of the form $\mathbf{x}\cdot\mathbf{e}_\ell$ for $\ell \in [k]$. By Yao's minimax principle [Yao77], there exists a deterministic algorithm $\mathcal{A}'$ with the same guarantee as $\mathcal{A}$ and the same expected cost over the input distribution. We then design an algorithm $\mathcal{B}$ using $\mathcal{A}'$ as follows. We independently sample a uniformly random index $i^* \sim \mathrm{Unif}([n])$ and $n-1$ points $\{\mathbf{x}_i\}_{i=1}^{n-1}$ with $\mathbf{x}_i \sim \mathrm{Unif}(\mathcal{S}_k)$ for each $i \in [n-1]$. We then run $\mathcal{A}'$ on the kernel matrix instance $\mathbf{K}$ corresponding to setting the $i^*$th standard basis vector to $\mathbf{x}$ and the other $n-1$ vectors according to $\{\mathbf{x}_i\}_{i=1}^{n-1}$. Note then that we can generate any entry of $\mathbf{K}$ on row $i^*$ or column $i^*$ by an inner product query to $\mathbf{x}$, and otherwise we can simulate the kernel query without making any inner product queries to $\mathbf{x}$. If $\mathcal{A}'$ ever reads more than $k/100$ entries of $\mathbf{x}$, we output failure.

Note that with probability at least $99/100$ over $\{\mathbf{x}_i\}_{i=1}^{n-1}$, each $\mathbf{e}_j$ for $j \in [k]$ is drawn at least once for $n$ large enough. Thus, in this event, $\mathbf{K}$ has rank $k$ only if $\mathbf{x} \sim \mathrm{Unif}(\mathcal{S}_k)$ and $k+1$ otherwise. Since $\mathcal{A}'$ is correct with probability $5/8$, by a union bound, $\mathcal{B}$ is correct with probability at least $1 - (\frac{1}{100} + \frac{3}{8}) \geq 5/9$.

Let $S$ denote the random variable indicating the list of positions of $\mathbf{K}$ read by $\mathcal{A}'$ and its corresponding values,let $L_1$ denote the distribution of $S$ conditioned on $\mathbf{x} \sim \mathrm{Unif}(\mathcal{S}_k)$, and $L_2$ the distribution of $S$ conditioned on $\mathbf{x} = \mathbf{e}_{k+1}$. Then by Proposition 2.58 of [BYP02], we have that

$$\frac{1 + D_{TV}(L_1, L_2)}{2} \geq \frac{5}{9} \tag{4.1}$$

so $D_{TV}(L_1, L_2) \geq 1/9$.

Since $r = o(nk)$, by averaging, for at least a $199/200$ fraction of the $n$ rows of $\mathbf{K}$, $\mathcal{A}'$ reads at most $k/200$ entries of the row $\mathbf{e}_i^\top \mathbf{K}$. Similarly, for at least a $199/200$ fraction of the $n$ columns of $\mathbf{K}$, $\mathcal{A}'$ reads at most $k/200$ entries of the column $\mathbf{K}\mathbf{e}_i$. Thus, for at least a $99/100$ fraction of the input points, $\mathcal{A}'$ makes at most $k/100$ inner product queries.

If we condition on the event that $\mathbf{x} \sim \mathrm{Unif}(\mathcal{S}_k)$, we have that $\mathcal{A}'$ makes at most $k/100$ inner product queries with $\mathbf{x}$ with probability at least $99/100$ over the randomness of $i^*$ by symmetry. That is, if $\mathcal{E}$ is the event that $\mathcal{A}'$ makes at most $k/100$ inner product queries on row and column $i^*$, then

$$\mathop{\mathbf{Pr}}_{\substack{i^* \sim \mathrm{Unif}([n]) \\ \{\mathbf{x}_i\}_{i=1}^{n-1} \sim \mathrm{Unif}(\mathcal{S}_k)^{n-1} \\ \mathbf{x} \sim \mathrm{Unif}(\mathcal{S}_k)}} (\mathcal{E}) \geq \frac{99}{100}. \tag{4.2}$$

Now letting $\mathcal{E}'$ be the event that $\mathcal{A}'$ sees 0s on all of its queries on row and column $i^*$, we have that

$$\mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \{\mathbf{x}_i\}_{i=1}^{n-1}\sim\mathrm{Unif}(\mathcal{S}_k)^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}' \mid \mathcal{E}) = \sum_{w\in\Omega} \mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}\left(\mathcal{E}' \mid \mathcal{E}, \{\mathbf{x}_i\}_{i=1}^{n-1} = w\right) \mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}\left(\mathcal{E}' \mid \mathcal{E}, \{\mathbf{x}_i\}_{i=1}^{n-1} = w\right). \tag{4.3}$$

Once we fix $\{\mathbf{x}_i\}_{i=1}^{n-1}$, there is a $1-1/k$ probability over $\mathbf{x}$ that any fixed query $\mathbf{x}\cdot\mathbf{e}_\ell$ returns a 0, so the probability that $\mathcal{E}'$ happens is at least

$$\left(1-\frac{1}{k}\right)^{k/100} \geq 1-\frac{1}{k}\frac{k}{100} = \frac{99}{100}. \tag{4.4}$$

Thus, by the chain rule,

$$\begin{aligned}\mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \{\mathbf{x}_i\}_{i=1}^{n-1}\sim\mathrm{Unif}(\mathcal{S}_k)^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}'\cap\mathcal{E}) &= \mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \{\mathbf{x}_i\}_{i=1}^{n-1}\sim\mathrm{Unif}(\mathcal{S}_k)^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}' \mid \mathcal{E}) \mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \{\mathbf{x}_i\}_{i=1}^{n-1}\sim\mathrm{Unif}(\mathcal{S}_k)^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}) \\ &\geq \frac{99}{100}\frac{99}{100} \geq \frac{49}{50}.\end{aligned} \tag{4.5}$$

Bounding the event $\mathcal{E}'\cup\mathcal{E}$ by $\mathcal{E}'$ as sets, we have that

$$\mathop{\mathbf{Pr}}_{\substack{i^*\sim\mathrm{Unif}([n])\\ \{\mathbf{x}_i\}_{i=1}^{n-1}\sim\mathrm{Unif}(\mathcal{S}_k)^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}') \geq \frac{49}{50}. \tag{4.6}$$

Also let $W \subseteq \mathrm{supp}(S)$ denote the set of $s\in\mathrm{supp}(S)$ such that $\mathcal{A}'$ reads all zeros on row and column $i^*$. Then,

$$\begin{aligned} D_{TV}(L_1,L_2) &= \sum_{s\in\mathrm{supp}(S)} \left| \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(S=s) - \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}=\mathbf{e}_{k+1}}}(S=s)\right| \\ &= \sum_{s\in\mathrm{supp}(S)} \left| \left( \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(S=s\mid\mathcal{E}') \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}') - \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}=\mathbf{e}_{k+1}}}(S=s)\right)\right| \\ &= \sum_{s\in\mathrm{supp}(S)} \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}=\mathbf{e}_{k+1}}}(S=s) \left| \mathop{\mathbf{Pr}}_{\substack{i^*,\{\mathbf{x}_i\}_{i=1}^{n-1}\\ \mathbf{x}\sim\mathrm{Unif}(\mathcal{S}_k)}}(\mathcal{E}') - 1\right| \leq \frac{1}{50}. \end{aligned} \tag{4.7}$$

This contradicts the conclusion that $D_{TV}(L_1,L_2)\geq 1/9$ so we conclude as desired. □

With the lemma in hand, we prove the $\Omega(nk)$ lower bound for approximating the cost of KKMC.

**Proposition 4.3.** *Let $k^2 = O(n)$. Suppose there exists an algorithm which, with probability at least $2/3$ over its random coin tosses and random inner product matrix $\mathbf{K}\sim\mu_{\mathrm{rank}}(n,k)$, correctly computes the optimal cost of the kernel $k$-means clustering instance up to a $(1\pm 1/2)$ relative error. Furthermore, suppose that the algorithm reads at most $r$ positions of $\mathbf{K}$, possibly adaptively. Then, $r=\Omega(nk)$.*

*Proof.* Assume that $nk = o(n^2)$, since otherwise the lower bound is best possible. As in the proof of Theorem 3.1, we have by Chernoff bounds that the number of $\mathbf{e}_j$ drawn is $(1\pm\frac{1}{100})n/k$ with probability at least $99/100$ for $n/k$ large enough for all $j\in[k]$ simultaneously. Note then that the optimal cost when $\mathbf{x}\sim\mathrm{Unif}(\{\mathbf{e}_j : j\in[k]\})$ is 0, since we can just take the centers to each be $\mathbf{e}_j$ for $j\in[k]$. On the other hand, when $\mathbf{x}=\mathbf{e}_{k+1}$, then there are more than $k$ types of vectors and thus the cost cannot be 0.

Thus, with probability $99/100$, the algorithm must correctly distinguish the two cases whenever the algorithm correctly approximates the optimal cost up to $(1\pm 1/2)$ relative error. Since the algorithm does this with probability $2/3$, by the union bound, the overall algorithm succeeds with probability at least $1-(\frac{1}{100}+\frac{1}{3})\geq 5/8$. Thus, the algorithm reads $\Omega(nk)$ kernel entries by Lemma 4.2. □

## 4.2 Main lower bound

### 4.2.1 The construction

We describe our hard input distribution $\mu_{\mathrm{KKMC}}(n,k,\varepsilon)$, formed as an inner product matrix of points drawn from the ambient space $\mathbb{R}^{k/\varepsilon}$.

**Definition 4.4** (Hard input distribution – kernel $k$-means clustering)**.** *Let $\varepsilon > 0, k, n$ be such that $k\binom{\varepsilon^{-1}}{2} = o(n)$ and $k/\varepsilon = \omega(1)$. We first define a distribution $\nu_{\mathrm{KKMC}}(k,\varepsilon)$ over vectors in $\mathbb{R}^{k/\varepsilon}$ as follows. First divide the $k/\varepsilon$ coordinates into $k$ blocks of $1/\varepsilon$ dimensions. Then, we sample our point set as follows: first uniformly select some block $j \in [k]$, and then uniformly select one of the $\binom{1/\varepsilon}{2}$ pairs $(j_1, j_2)$ where $j_1, j_2 \in [1/\varepsilon]$ with $j_1 \neq j_2$, and then output $\mathbf{v}_{j,j_1,j_2} := (\mathbf{e}_{\ell_1} + \mathbf{e}_{\ell_2})/\sqrt{2}$, where $\ell_1 = j/\varepsilon + j_1, \ell_2 = j/\varepsilon + j_2$. We then generate an i.i.d. sample $\{\mathbf{x}_i\}_{i=1}^n$ of $n$ points drawn from $\nu_{\mathrm{KKMC}}(k,\varepsilon)$ and then generate $\mathbf{K} \sim \mu_{\mathrm{KKMC}}(n,k,\varepsilon)$ by setting it to be the inner product matrix of $\{\mathbf{x}_i\}_{i=1}^n$, i.e. $\mathbf{e}_i^\top \mathbf{K} \mathbf{e}_j := \mathbf{x}_i \cdot \mathbf{x}_j$. For $\mathbf{x}$ in the support of $\nu_{\mathrm{KKMC}}(k,\varepsilon)$, we let* $\mathrm{block}(\mathbf{x})$ *denote the $j \in [k]$ such that $\mathbf{x} = \mathbf{v}_{j,j_1,j_2}$.*

Intuitively, we are adding "edges" between pairs of coordinates in the same block of $1/\varepsilon$ coordinates, so that clusterings that associate points in the same block together have lower cost.

In this section, we will prove the following main theorem:

**Theorem 4.5** (Query lower bound for kernel $k$-means clustering)**.** *Let $\varepsilon, k, n$ be such that $k\binom{\varepsilon^{-1}}{2} = o(n)$. Suppose an algorithm $\mathcal{A}$ finds a $(1 \pm \varepsilon)$-approximate solution to a kernel $k$-means clustering instance drawn from $\mu_{\mathrm{KKMC}}(n,k,\varepsilon)$ with probability at least $2/3$ over its random coin tosses and the input distribution. Then, $\mathcal{A}$ makes at least $\Omega(nk/\varepsilon)$ kernel queries.*

This lower bound is tight up to logarithmic factors, nearly matching for example the ridge leverage score algorithm of Theorem 16 in [MM17] which reads $O\big(\frac{nk}{\varepsilon}\log\frac{k}{\varepsilon}\big)$ kernel entries.

### 4.2.2 Proof overview

In our proof, we will think of any clustering as being divided into two groups: the points $S$, which are clustered to "dense" clusters, and the points $\overline{S}$, which are clustered to "sparse" clusters. Roughly, if we fix a point in a dense cluster and sample points randomly from that cluster, then we have a high probability (at least $\Omega(\varepsilon)$) of finding a point that has nonzero inner product with it. We then argue that if there are not enough points in dense clusters, then the cost of the clustering is too large, so the clustering cannot be a $(1+\varepsilon)$-approximate solution to the optimal kernel $k$-means clustering solution. Then, we show that finding a lot of points in dense clusters can solve a computational problem that requires $\Omega(nk/\varepsilon)$ kernel queries, which then yields Theorem 4.5.

The main work that needs to be done is lower bounding the cost of clustering the points that belong to dense clusters, since it is easy to see that sparse clusters have high cost. Among the dense clusters, if the size of the cluster is at least $n/k$, which we call the "large" clusters, then the cost is easy to bound. The worrisome part is the "small" clusters, which have the potential of having very small cost per point. We will show that if we carefully bound the cost of small clusters as a function of their size, then if we don't have too many points total that belong to small clusters, then the cost is still high enough to achieve the desired result.

## 4.3 Cost computations

### 4.3.1 The cost of a good clustering

Consider the clustering that assigns all points supported in the same block with each other. We first do our cost computations for the average case, where every vector $\mathbf{v}_{j,j_1,j_2}$ is drawn the same number of times. Then, the first block has center

$$\frac{1}{\binom{\varepsilon^{-1}}{2}} \sum_{(i,j)\in\binom{[\varepsilon^{-1}]}{2}} \frac{\mathbf{e}_i + \mathbf{e}_j}{\sqrt{2}} = \frac{(\varepsilon^{-1}-1)}{\sqrt{2}\binom{\varepsilon^{-1}}{2}} \sum_{i\in[\varepsilon^{-1}]} \mathbf{e}_i = \sqrt{2}\varepsilon \sum_{i\in[\varepsilon^{-1}]} \mathbf{e}_i \tag{4.8}$$

and the center for the rest of the blocks is similar. Then, the cost of a single point $(\mathbf{e}_{i^*} + \mathbf{e}_{j^*})/\sqrt{2}$ is

$$\left\|\frac{\mathbf{e}_{i^*} + \mathbf{e}_{j^*}}{\sqrt{2}} - \sqrt{2}\varepsilon \sum_{i\in[\varepsilon^{-1}]} \mathbf{e}_i\right\|_2^2 = 2\left(\frac{1}{\sqrt{2}} - \sqrt{2}\varepsilon\right)^2 + \left(\varepsilon^{-1} - 2\right)\left(\sqrt{2}\varepsilon\right)^2 = 1 - 4\varepsilon + \varepsilon^{-1}2\varepsilon^2 - 4\varepsilon^2 = 1 - 2\varepsilon - 4\varepsilon^2. \tag{4.9}$$

Thus, the cost of this clustering is like $n(1-2\varepsilon)$. Note that this computation also gives a lower bound on the cost of a cluster containing $n/k$ points, since for any cluster of size $n/k$, we can clearly improve its cost while we can swap points to be supported on the same block.

Now by Chernoff bounds, with probability tending to 1 as $n/k\binom{\varepsilon^{-1}}{2}$ tends to infinity, the cost of this clustering is bounded above by

$$n\left(1 - \left(1 - \frac{1}{40}\right)2\varepsilon\right) = n\left(1 - \frac{79}{40}\varepsilon\right). \tag{4.10}$$

and the cost of any cluster of size $n/k$ is bounded below by

$$\frac{n}{k}\left(1 - \left(1 + \frac{1}{40}\right)2\varepsilon\right) = \frac{n}{k}\left(1 - \frac{81}{40}\varepsilon\right). \tag{4.11}$$

This proves the following lemmas.

**Lemma 4.6** (Cost bound for an optimal clustering)**.** *With probability at least* $99/100$*, the cost of an optimal clustering is at most* $n(1-(79/40)\varepsilon)$*.*

**Lemma 4.7** (Cost bound for a large cluster)**.** *Let $C$ be a cluster of size at least $n/k$. Then with probability at least* $99/100$*, the cost per point of $C$ is bounded below by* $1-(81/40)\varepsilon$*.*

### 4.3.2 The cost of a small cluster

We will first prove a lower bound on the cost of a fixed cluster $C$. In this section, we will parameterize our lower bound only by the size of the cluster, and then later use this result to lower bound the cost of any clustering. When we prove our lower bound result, it will be useful to introduce the quantities

$$\alpha \coloneqq \frac{n}{k\binom{\varepsilon^{-1}}{2}}, \qquad \tau \coloneqq \frac{|C|}{\alpha}. \tag{4.12}$$

Intuitively, $\alpha$ is the number of copies of a vector $\mathbf{v}_{j,j_1,j_2}$ we expect to draw, and $\tau$ is the minimal number of different types of vectors $\mathbf{v}_{j,j_1,j_2}$ we can have in our cluster $C$. Note that by the union and Chernoff bounds, there are $(1\pm\gamma)\alpha$ copies of each vector with probability $k\binom{\varepsilon^{-1}}{2}\exp\left(-\gamma^2\alpha/3\right)$, where $\gamma$ is a small constant to be chosen. The lower bound we prove then is the following.

**Lemma 4.8** (Cost bound for a small cluster)**.** *Let $\gamma$ be a constant small enough so that*

$$(1+\gamma)^2\frac{(1+2\sqrt{\gamma})^2}{(1-2\sqrt{\gamma})^3} \leq 1 + \frac{1}{20}. \tag{4.13}$$

*Let $\alpha$ and $\tau$ be defined as above, with respect to a cluster $C$ of points drawn from $\nu_{\mathrm{KKMC}}(k,\varepsilon)$ with size bounded by*

$$\frac{\alpha}{\gamma} = \Theta(\alpha) \leq |C| \leq \frac{n}{k} \tag{4.14}$$

*Additionally, define the quantity*

$$\kappa \coloneqq (\varepsilon^{-1} - 1/2) - \sqrt{(\varepsilon^{-1} - 1/2)^2 - 2\tau}. \tag{4.15}$$

*Then, the cost on this cluster is bounded below by*

$$\mathrm{cost}(C) \geq |C| - \left(1 + \frac{1}{20}\right)\frac{\alpha}{2}\frac{\kappa(\varepsilon^{-1}-1)^2 + \kappa^2(\varepsilon^{-1}-\kappa)}{\tau}. \tag{4.16}$$

**Remark 4.9.** *Note that if a cluster has size at most $\alpha$, then its cost can be 0 by taking them all to be the same vector, and here, we only require the cluster to have size a constant times $\alpha$ to get the lower bound.*

*Proof.* To prove this lemma, we will first reduce to the case of considering clusters which have all copies of all of its vectors, and then conclude the lower bound by optimizing the cost of clusters of this form.

**Reduction to maximizing a sum of squares.** Since we wish to lower bound the cost of this cluster, we assume that it is supported on $\varepsilon^{-1}$ coordinates since using more than $\varepsilon^{-1}$ coordinates is clearly suboptimal, and we make take the entire cluster to be drawn from one block by the upper bound on the size of $C$. Then for $i \in [\varepsilon^{-1}]$, let $n_i$ denote the number of vectors supported on the $i$th coordinate, so $\sum_{i\in[\varepsilon^{-1}]} n_i = 2|C|$. Let $\boldsymbol{\mu}$ denote the center of $C$. Then, it's easy to see that

$$\boldsymbol{\mu} = \frac{1}{|C|}\sum_{i\in[\varepsilon^{-1}]} \frac{n_i}{\sqrt{2}}\mathbf{e}_i. \tag{4.17}$$

Thus, for each point $\mathbf{v}_{i,j} := (\mathbf{e}_i + \mathbf{e}_j)/\sqrt{2}$ in $C$, its cost contribution is

$$\left(\frac{1}{\sqrt{2}} - \boldsymbol{\mu}_i\right)^2 + \left(\frac{1}{\sqrt{2}} - \boldsymbol{\mu}_j\right)^2 + \sum_{\ell\in[\varepsilon^{-1}]\setminus\{i,j\}} (0-\boldsymbol{\mu}_\ell)^2 = 1 + \|\boldsymbol{\mu}\|_2^2 - \sqrt{2}(\boldsymbol{\mu}_i + \boldsymbol{\mu}_j). \tag{4.18}$$

Then, the total cost for the cluster is

$$\begin{aligned}\sum_{\mathbf{v}_{i,j}\in C} 1 + \|\boldsymbol{\mu}\|_2^2 - \sqrt{2}(\boldsymbol{\mu}_i + \boldsymbol{\mu}_j) &= \sum_{\mathbf{v}_{i,j}\in C} 1 + \|\boldsymbol{\mu}\|_2^2 - \sqrt{2}\left(\frac{n_i}{\sqrt{2}|C|} + \frac{n_j}{\sqrt{2}|C|}\right) \\ &= |C|(1 + \|\boldsymbol{\mu}\|_2^2) - \frac{1}{|C|}\sum_{\mathbf{v}_{i,j}\in C} n_i + n_j \\ &= |C|\left(1 + \sum_{i\in[\varepsilon^{-1}]}\frac{n_i^2}{2|C|^2}\right) - \frac{1}{|C|}\sum_{i\in[\varepsilon^{-1}]} n_i^2 \\ &= |C| - \frac{1}{2|C|}\sum_{i\in[\varepsilon^{-1}]} n_i^2.\end{aligned} \tag{4.19}$$

Thus, going forward, we will forget about the $|C|$ and focus only on the sum of squared counts. In the following arguments, we may give the impression of adding vectors to the cluster to bound this quantity above, but we will do this without touching this $|C|$ term.

**Discretization.** Consider two vectors $\mathbf{v}_{i_1,i_2}, \mathbf{v}_{j_1,j_2}$ where $n_{i_1} + n_{i_2} \geq n_{j_1} + n_{j_2}$. Then, we have that

$$\begin{aligned}(n_{i_1}+1)^2 + (n_{i_2}+1)^2 + (n_{j_1}-1)^2 + (n_{j_2}-1)^2 &= n_{i_1}^2 + n_{i_2}^2 + n_{j_1}^2 + n_{j_2}^2 + 4 + 2(n_{i_1} + n_{i_2} - n_{j_1} - n_{j_2}) \\ &> n_{i_1}^2 + n_{i_2}^2 + n_{j_1}^2 + n_{j_2}^2\end{aligned} \tag{4.20}$$

so we can strictly improve the cost of any clustering by changing a vector with low total count of coordinates to one with higher total count of coordinates. Thus, we may reduce our lower bound proof to the case where every vector type $\mathbf{v}_{i,j}$ has every copy of itself, except for possibly one "leftover" vector type. By adding in all copies of this leftover vector type, which only makes the sum of squared counts larger, we assume that every vector type has every copy of itself.

**Filling up coordinates.** We do a similar argument as in the above to further constrain the form of the clusters we consider. Note that

$$(n_i+1)^2 + (n_j-1)^2 = n_i^2 + n_j^2 + 2 + 2(n_i - n_j) > n_i^2 + n_j^2 \tag{4.21}$$

for $n_i \geq n_j$, so we improve the clustering by iteratively swapping vectors $v_{i,j}$ to $v_{i^*,j}$ where $i^*$ is the coordinate with the largest count $n_{i^*}$ that still hasn't exhausted the $(\varepsilon^{-1} - 1)$ different types of vectors $v_{i^*,j}$ for $j \in [\varepsilon^{-1}] \setminus \{j\}$ that is supported on the $i^*$th coordinate. By relabeling the coordinates if necessary, WLOG assume that we fill up the coordinates $i = 1, 2, \ldots, \varepsilon^{-1}$ in order.

On the first coordinate, we can use $(\varepsilon^{-1} - 1)$ types of vectors to fill it all the way up to

$$n_1 = (\varepsilon^{-1} - 1)(1 \pm \gamma)\alpha = (1 \pm \gamma)(\varepsilon^{-1} - 1)\frac{n}{k\binom{\varepsilon^{-1}}{2}} = (1 \pm \gamma)\frac{2n\varepsilon}{k} \tag{4.22}$$

vectors, where the $(\varepsilon^{-1} - 1)$ types of vectors are $(\mathbf{e}_1 + \mathbf{e}_j)/\sqrt{2}$ for $j \in [\varepsilon^{-1}] \setminus \{1\}$. Note that at this point, every other coordinate will have $(1 \pm \gamma)\alpha$ vectors. Then we need to maximize the second coordinate, which already has 1 type and $(1 \pm \gamma)\alpha$ vectors, so we can use $(\varepsilon^{-1} - 2)$ additional types of vectors to fill it all the way up, at which point all the other coordinates will have $2(1 \pm \gamma)\alpha$ vectors. We can do this until we fill up $K$ coordinates at which point we have used $T$ types of vectors. Note that we can solve for $K$ by solving

$$\begin{aligned} T = \sum_{i=1}^{K} \varepsilon^{-1} - i = K\varepsilon^{-1} - \frac{K(K+1)}{2} &\iff \frac{1}{2}K^2 + \left(\frac{1}{2} - \varepsilon^{-1}\right)K + T = 0 \\ &\iff K = (\varepsilon^{-1} - 1/2) \pm \sqrt{(\varepsilon^{-1} - 1/2)^2 - 2T}. \end{aligned} \tag{4.23}$$

Since we can only have $\varepsilon^{-1}$ coordinates, we need to take the solution

$$K = (\varepsilon^{-1} - 1/2) - \sqrt{(\varepsilon^{-1} - 1/2)^2 - 2T}. \tag{4.24}$$

**Bounding.** Note that from filling up the last leftover vector type, we have that $T \leq \lceil |C|/(1-\gamma)\alpha \rceil$ and $T \geq \lfloor |C|/(1+\gamma)\alpha \rfloor$. Thus, $|C| = (1 \pm \gamma)\alpha(T \pm 1)$. Now note that we require that $\alpha/\gamma \leq |C|$ in equation (4.14), which means that

$$\tau = \frac{|C|}{\alpha} \geq \frac{\alpha/\gamma}{\alpha} = \frac{1}{\gamma} \implies \tau\gamma \geq 1 \tag{4.25}$$

and thus $T = (1 \pm \gamma)\tau \pm 1 = (1 \pm 2\gamma)\tau$. It then follows that for $\kappa$ defined as in (4.15),

$$\begin{aligned} K &= (\varepsilon^{-1} - 1/2) - \sqrt{(\varepsilon^{-1} - 1/2)^2 - 2(1 \pm 2\gamma)\tau} \\ &= \left[(\varepsilon^{-1} - 1/2) - \sqrt{(\varepsilon^{-1} - 1/2)^2 - 2\tau}\right] \pm \sqrt{4\gamma\tau} \\ &= \kappa \pm \sqrt{4\gamma\binom{\kappa}{2}} = (1 \pm 2\sqrt{\gamma})\kappa \end{aligned} \tag{4.26}$$

For $K$ coordinates we fill the coordinates all the way up to $(\varepsilon^{-1}-1)(1\pm\gamma)\alpha$ and otherwise we have $K(1\pm\gamma)\alpha$ vectors. Then, the sum of squared counts is bounded by

$$\begin{aligned} \sum_{i \in [\varepsilon^{-1}]} n_i^2 &\leq K\big((\varepsilon^{-1} - 1)(1+\gamma)\alpha\big)^2 + (\varepsilon^{-1} - K)\big(K(1+\gamma)\alpha\big)^2 \\ &= ((1+\gamma)\alpha)^2\big[K(\varepsilon^{-1} - 1)^2 + K^2(\varepsilon^{-1} - K)\big] \end{aligned} \tag{4.27}$$

so the cost on this cluster is at least

$$\begin{aligned} |C| - \frac{1}{2|C|}\sum_{i \in [\varepsilon^{-1}]} n_i^2 &\geq |C| - \frac{1}{2|C|}((1+\gamma)\alpha)^2\big[K(\varepsilon^{-1} - 1)^2 + K^2(\varepsilon^{-1} - K)\big] \\ &= |C| - \left(1 + \frac{1}{20}\right)\frac{\alpha}{2}\frac{\kappa(\varepsilon^{-1} - 1)^2 + \kappa^2(\varepsilon^{-1} - \kappa)}{\tau} \end{aligned} \tag{4.28}$$

since we chose $\gamma$ to be small enough in equation (4.13). □

### 4.3.3 Optimizing over $k$ clusters

In Lemma 4.8, we have found a lower bound on the cost of a fixed cluster that only makes reference to the size of the cluster. All that is left to do is to optimize the sum of these functions under the constraint of the total number of points to cluster. Recall from the proof overview that we are in the context of lower bounding the cost of a subset of the points $S$, which we wish to show must be large. Thus, we will assume that $|S| \leq 2n/5$ in this lower bound. The formal statement of the lemma we prove here is the following:

**Lemma 4.10** (Cost bound for $\ell$ clusters)**.** *Suppose $S$ is a set of at most $|S| \leq 2n/5$ points drawn from $\nu_{\mathrm{KKMC}}(k,\varepsilon)$. Then, for any clustering $\mathcal{C}_S$ of $S$ into $\ell \leq k$ clusters,*

$$\mathrm{cost}(\mathcal{C}_S) \geq |S| - \frac{77}{40}n\varepsilon. \tag{4.29}$$

*Proof.* Note that we only decrease the cost of a clustering if we allow a single cluster to be split up into multiple clusters. Then for any "large" cluster $C$ with size at least $|C| > n/k$, we can treat every $n/k$ points as a separate cluster and lower bound its cost at $1-2\varepsilon$ per point. Then, let $S = \mathcal{S} \cup \mathcal{T}$, where $\mathcal{S}$ is the set of clusters that now belong to clusters of size at most $n/k$ and $\mathcal{T}$ is the set of points whose cost we have bounded below by $1-(81/40)\varepsilon$ by Lemma 4.7.

Recall that our lower bound for a single cluster, Lemma 4.8, only applies to clusters of size at least $\alpha/\gamma$. Let $\mathcal{L} \subseteq [\ell]$ be the set of indices of clusters such that the lower bound applies and let $\mathcal{M} \subseteq \mathcal{S}$ be the set of all points in such a cluster. Now applying this lower bound for these clusters and lower bounding by 0 for the others, we arrive at the lower bound

$$\mathrm{cost}(\mathcal{C}_S) \geq |\mathcal{T}|\left(1 - \frac{81}{40}\varepsilon\right) + \sum_{i\in\mathcal{L}} |C_i| - \left(1+\frac{1}{20}\right)\frac{\alpha}{2}\frac{\kappa_i(\varepsilon^{-1}-1)^2 + \kappa_i^2(\varepsilon^{-1}-\kappa_i)}{\tau_i}. \tag{4.30}$$

Together with the constraint that there are $|\mathcal{M}|$ points for which we can apply the lower bound of Lemma 4.8, we now focus on the optimization problem

$$\begin{aligned} \text{minimize} \quad & |\mathcal{M}| - \left(1+\frac{1}{20}\right)\frac{\alpha}{2}\sum_{i\in\mathcal{L}} \frac{\kappa_i(\varepsilon^{-1}-1)^2 + \kappa_i^2(\varepsilon^{-1}-\kappa_i)}{\tau_i} \\ \text{subject to} \quad & \alpha\sum_{i\in\mathcal{L}}\tau_i = |\mathcal{M}|, \qquad 0 \leq \tau_i \leq \binom{\varepsilon^{-1}}{2} = \frac{(\varepsilon^{-1}-1/2)^2}{2} - \frac{1}{8} \end{aligned} \tag{4.31}$$

where

$$\kappa_i = (\varepsilon^{-1}-1/2) - \sqrt{(\varepsilon^{-1}-1/2)^2 - 2\tau_i}. \tag{4.32}$$

We now introduce less cumbersome notation to get our bound. Let

$$R \coloneqq (\varepsilon^{-1}-1/2), \qquad u_i \coloneqq R^2 - 2\tau_i. \tag{4.33}$$

Then by plugging in definitions, the optimization problem is now

$$\begin{aligned} \text{minimize} \quad & |\mathcal{M}| - \left(1+\frac{1}{20}\right)\alpha\sum_{i\in\mathcal{L}} \frac{(R-\sqrt{u_i})(R-1/2)^2 + (R-\sqrt{u_i})^2(\sqrt{u_i}+1/2)}{R^2-u_i} \\ \text{subject to} \quad & \sum_{i\in\mathcal{L}} u_i = |\mathcal{L}|R^2 - 2|\mathcal{M}|/\alpha, \qquad \frac{1}{4} \leq u_i \leq R^2 \end{aligned} \tag{4.34}$$

Now note (WolframAlpha link) that

$$\begin{aligned} \frac{(R-\sqrt{u_i})(R-1/2)^2 + (R-\sqrt{u_i})^2(\sqrt{u_i}+1/2)}{R^2-u_i} &= 2R - \frac{1}{2} - \left(\frac{4R^2-1}{4(R+\sqrt{u_i})} + \sqrt{u_i}\right) \\ &\leq 2R - \frac{1}{2} - \left(\frac{4R^2-1}{8R} + \sqrt{u_i}\right) = \frac{3}{2}R - \frac{1}{2} + \frac{1}{8R} - \sqrt{u_i}. \end{aligned} \tag{4.35}$$

Then, by noting that $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$, we can optimize

$$\begin{aligned} &\text{maximize} \quad \sum_{i\in\mathcal{L}} -\sqrt{u_i} \\ &\text{subject to} \quad \sum_{i\in\mathcal{L}} u_i = |\mathcal{L}|R^2 - 2|\mathcal{M}|/\alpha, \qquad \frac{1}{4} \le u_i \le R^2 \end{aligned} \tag{4.36}$$

by setting $u_i = R^2$ for $(|\mathcal{L}|R^2 - 2|\mathcal{M}|/\alpha)/R^2$ of the $i$ and the rest to 0 (the minimum $u_i$ is 1/4, but allowing it to be 0 is a relaxation since we optimize over a larger domain). Thus, the value of optimization problem (4.36) is at most

$$\frac{|\mathcal{L}|R^2 - 2|\mathcal{M}|/\alpha}{R^2}(-R) = -\left(|\mathcal{L}|R - \frac{2|\mathcal{M}|}{\alpha R}\right). \tag{4.37}$$

Thus, we have that

$$\alpha\left(\sum_{i\in\mathcal{L}} \frac{3}{2}R - \frac{1}{2} + \frac{1}{8R} - \sqrt{u_i}\right) \le \alpha|\mathcal{L}|\left(\frac{3}{2}R - \frac{1}{2} + \frac{1}{8R}\right) - \alpha\left(|\mathcal{L}|R - \frac{2|\mathcal{M}|}{\alpha R}\right). \tag{4.38}$$

Now recall that we set $|\mathcal{M}| + |\mathcal{T}| \le |S| \le 2n/5$ and $|\mathcal{L}| \le k$ so the above is bounded above by

$$\begin{aligned} \alpha|\mathcal{L}|\left(\frac{3}{2}R - \frac{1}{2} + \frac{1}{8R}\right) - \alpha\left(|\mathcal{L}|R - \frac{2|\mathcal{M}|}{\alpha R}\right) &= \frac{1}{2}\alpha|\mathcal{L}|R - \alpha|\mathcal{L}|\left(\frac{1}{2} - \frac{1}{8R}\right) + \alpha\frac{2|\mathcal{M}|}{\alpha R} \\ &\le \frac{1}{2}\alpha kR - \alpha|\mathcal{L}|\left(\frac{1}{2} - \frac{1}{8R}\right) + \frac{2}{5}\alpha kR - \frac{2|\mathcal{T}|}{R} \\ &= \frac{9}{10}\alpha kR - \alpha|\mathcal{L}|\left(\frac{1}{2} - \frac{1}{8R}\right) - \frac{2|\mathcal{T}|}{R} \\ &\le \frac{9}{10}\alpha kR - 2\varepsilon|\mathcal{T}|. \end{aligned} \tag{4.39}$$

Thus, we may lower bound the value of the optimization problem of (4.31) by

$$|\mathcal{M}| - \left(1 + \frac{1}{20}\right)\left(\frac{9}{10}\alpha kR - 2\varepsilon|\mathcal{T}|\right). \tag{4.40}$$

Now finally, note that there are at most $k$ clusters where the lower bound of Lemma 4.8 doesn't apply, so there are at most

$$|\mathcal{S} \setminus \mathcal{M}| \le k \cdot \frac{\alpha}{\gamma} = \Theta(n\varepsilon^2) \le \frac{1}{100}n\varepsilon \tag{4.41}$$

points that we have ignored the cost for, for $\varepsilon$ smaller than some constant. Collecting our bounds of (4.40) and (4.41) and plugging our definitions of $\alpha$ and $R$ back in, we obtain a lower bound of

$$\begin{aligned} \operatorname{cost}(\mathcal{C}_S) &\ge |\mathcal{T}|\left(1 - \frac{81}{40}\varepsilon\right) + |\mathcal{M}| - \left(1 + \frac{1}{20}\right)\frac{9}{10}\alpha kR + \left(1 + \frac{1}{20}\right)2\varepsilon|\mathcal{T}| \\ &= (|\mathcal{T}| + |\mathcal{M}|) + \left(\frac{42}{20} - \frac{81}{40}\right)\varepsilon|\mathcal{T}| - \left(1 + \frac{1}{20}\right)\frac{9}{10}\alpha kR \\ &\ge (|S| - |\mathcal{S} \setminus \mathcal{M}|) - \left(1 + \frac{1}{20}\right)\frac{9}{10}\alpha kR \\ &\ge |S| - \frac{1}{100}n\varepsilon - \left(1 + \frac{1}{20}\right)\frac{9}{5}n\varepsilon > |S| - \frac{77}{40}n\varepsilon \end{aligned} \tag{4.42}$$

on the cost of the clustering, as desired. □

## 4.4 Hardness

### 4.4.1 Cost lemma

We now prove a lemma that translates our cost computations from the above section into a statement about the probability of sampling nonzero inner products. Intuitively, we will prove that an approximately optimal solution to the kernel $k$-means clustering instance must output a clustering such that at least $2n/5$ of the points belong to a cluster with lots of points that share a coordinate with the point, i.e. "neighbors".

**Lemma 4.11** (Sampling probability of an approximate solution)**.** *Suppose that $\mathcal{C}$ is a $(1+\varepsilon/40)$-approximate solution to a kernel $k$-means clustering instance drawn as $\mathbf{K} \sim \mu_{\mathrm{KKMC}}(n, k, \varepsilon)$. Then for at least $2n/5$ of the points, if we uniformly sample dot products between the point and other points in its cluster, then there is at least an $\varepsilon/80$ probability of sampling a point that has nonzero inner product with the point.*

*Proof.* Suppose for contradiction that there are at most $2n/5$ points belonging to a cluster such that sampling uniformly from the cluster yields at least an $\varepsilon/80$ probability of sampling a point that has nonzero inner product with that point, which we refer to as a *neighbor*. Let $S$ be the set of points that belong to such a cluster with at least probability $\varepsilon/80$ of sampling a neighbor, and let $\overline{S}$ be the complement. We first compute the cost of a point $(\mathbf{e}_i + \mathbf{e}_j)/\sqrt{2}$ in $\overline{S}$. Let $C$ be the point's cluster and let $n_i, n_j$ be the number of points in the cluster that has support on the $i$th coordinate. Then, $n_i/|C|$ and $n_j/|C|$ are both at most $\varepsilon/80$. Now note that the $i$ and $j$th coordinates of the center are $n_i/(\sqrt{2}|C|)$ and $n_j/(\sqrt{2}|C|)$, so the cost of that point is at least

$$\left(\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}}\frac{n_i}{|C|}\right)^2 + \left(\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}}\frac{n_j}{|C|}\right)^2 \geq 1 - \frac{1}{40}\varepsilon. \tag{4.43}$$

Then $|S| \leq 2n/5$, so we may use the bounds from Lemma 4.10 to bound the cost from below by

$$|\overline{S}|\left(1 - \frac{1}{40}\varepsilon\right) + |S| - \frac{77}{40}n\varepsilon \geq n\left(1 - \frac{78}{40}\varepsilon\right). \tag{4.44}$$

Now recall that by Lemma 4.6, the optimal solution has cost at most $n(1 - (79/40)\varepsilon)$, so a $(1 + \varepsilon/40)$-approximate solution needs to have cost at most

$$n\left(1 - \frac{79}{40}\varepsilon\right)\left(1 + \frac{1}{40}\varepsilon\right) = n\left(1 - \frac{78}{40}\varepsilon - \frac{79}{1600}\varepsilon^2\right) < n\left(1 - \frac{78}{40}\varepsilon\right) \tag{4.45}$$

which the above solution does not. □

### 4.4.2 Hardness reduction

Finally, we give the hardness result. Consider the following computational problem LabelKKMC. Recall the definition of $\nu_{\mathrm{KKMC}}$ and block from Definition 4.4.

**Definition 4.12** (LabelKKMC)**.** *We first sample $n$ points $\{\mathbf{x}_i\}_{i=1}^n$ from our hard point set $\nu_{\mathrm{KKMC}}(k, \varepsilon)$, label the identity of the first $n/2$ points, and then ask an algorithm to correctly give* $\mathrm{block}(\mathbf{x}_i)$ *for $1/6$ of the remaining $n/2$ points.*

We will show that this problem requires reading $\Omega(nk/\varepsilon)$ kernel entries and that an algorithm solving the KKMC problem on this instance can solve this problem. We first prove the following lemma:

**Definition 4.13** (LabelSingleKKMC)**.** *Given as input $\mathbf{x} \sim \nu_{KKMC}(k, \varepsilon)$, determine* $\mathrm{block}(\mathbf{x})$.

**Lemma 4.14** (Hardness of LabelSingleKKMC)**.** *Let $\log_2 k \geq 12$. Suppose there exists an algorithm $\mathcal{A}$, possibly randomized, that correctly solves* LabelSingleKKMC *with probability at least $1/100$ over $\mathbf{x} \sim \nu_{\mathrm{KKMC}}(k, \varepsilon)$ and its random coin tosses. Furthermore, suppose that $\mathcal{A}$ makes at most $q$ inner product queries of the form $\mathbf{x} \cdot \mathbf{v}_{\ell,\ell_1,\ell_2}$, possibly adaptively, on any input. Then, $q \geq (k/\varepsilon)/100$.*

*Proof.* By way of Yao's minimax principle [Yao77], assume that the algorithm is deterministic. Note first that for a single $\mathbf{v}_{j,j_1,j_2}$, its inner product with any vector drawn from a different block is 0. Within the same block, its inner product with another $\mathbf{v}_{j,j_1',j_2'}$ is 0 as well, unless $j_1 = j_1'$ or $j_2 = j_2'$. This happens with probability

$$1 - \frac{\binom{\varepsilon^{-1}-2}{2}}{\binom{\varepsilon^{-1}}{2}} = \frac{\varepsilon^{-1}(\varepsilon^{-1}-1) - (\varepsilon^{-1}-2)(\varepsilon^{-1}-3)}{\varepsilon^{-1}(\varepsilon^{-1}-1)} = \frac{4\varepsilon^{-1}-6}{\varepsilon^{-1}(\varepsilon^{-1}-1)} = \frac{4\varepsilon - 6\varepsilon^2}{1-\varepsilon} \leq 8\varepsilon \tag{4.46}$$

for $\varepsilon \leq 1/2$ and thus for a fixed $\mathbf{v}_{\ell,\ell_1,\ell_2}$, we have that

$$\mathop{\mathbf{Pr}}_{\mathbf{x}\sim\nu_{\mathrm{KKMC}}(k,\varepsilon)}(\mathbf{x}\cdot\mathbf{v}_{\ell,\ell_1,\ell_2} \neq 0) \leq \frac{8\varepsilon}{k}. \tag{4.47}$$

We now use the above to first show a lower bound of $q = \Omega(k/\varepsilon)$ on the number of adaptive inner product queries $\mathbf{x}\cdot\mathbf{v}_{\ell,\ell_1,\ell_2}$ required to find $\mathrm{block}(\mathbf{x})$ for a single draw $\mathbf{x} \sim \nu_{\mathrm{KKMC}}(k,\varepsilon)$.

Assume for simplicity that $k$ is a power of 2, and for each $m \in [\log_2 k]$, consider the hypothesis test $\mathcal{H}_m$ of deciding whether the $m$th bit of $\mathrm{block}(\mathbf{x}) \in [k]$ is 0 or 1. Note that since we choose $j \in [k]$ uniformly, each of the $\log_2 k$ hypothesis tests are independent and identical and thus the error probability of the hypothesis test of the optimal success probability is the same for each hypothesis test. Note that an algorithm succeeds in correctly outputting $\mathrm{block}(\mathbf{x})$ if and only if it succeeds on all $\log_2 k$ of the hypothesis tests. If the optimal success probability is at most $2/3$, then for $\log_2 k \geq 12$, we have that the success probability on all $\log_2 k$ of the hypothesis tests is at most

$$\left(\frac{2}{3}\right)^{\log_2 k} \leq \left(\frac{2}{3}\right)^{12} < \frac{1}{100} \tag{4.48}$$

which means it does not have the required guarantees. Thus, $\mathcal{A}$ must solve each hypothesis test $\mathcal{H}_m$ with probability at least $2/3$.

Now fix a hypothesis test $\mathcal{H}_m$ from the above, let $\mathcal{E}_0$ be the event that the $m$th bit of $\mathrm{block}(\mathbf{x})$ is 0, and let $\mathcal{E}_1$ be the event that the $m$th bit of $\mathrm{block}(\mathbf{x})$ is 1. Let $S$ be the random variable indicating the list of positions of $\mathbf{K}$ read by $\mathcal{A}$ and its corresponding values, let $L_0$ denote the distribution of $S$ conditioned on $\mathcal{E}_0$, and let $L_1$ denote the distribution of $S$ conditioned on $\mathcal{E}_1$. Then by Proposition 2.58 of [BYP02], we have that

$$\frac{1 + D_{TV}(L_0, L_1)}{2} \geq \frac{2}{3} \tag{4.49}$$

so $D_{TV}(L_0, L_1) \geq 1/3$.

Now suppose for contradiction that $\mathcal{A}$ makes $q \leq (k/\varepsilon)/100$ queries on any input. Since $\mathcal{A}$ is deterministic, it makes the same sequence of inner product queries if it reads a sequence of $q$ zeros. For a fixed query $\mathbf{v}_{\ell,\ell_1,\ell_2}$, the probability that $\mathbf{x}\cdot\mathbf{v}_{\ell,\ell_1,\ell_2} \neq 0$ is at most $8\varepsilon/k$ by equation (4.47). Then by a union bound over the $q$ queries, the probability that the algorithm seems any zeros is at most $8(\varepsilon/k)q \leq 8/100$. Then, letting $\Omega$ denote the support of $S$ and $s_0$ the value of $S$ when $\mathcal{A}$ reads all zeros, we have that

$$\begin{aligned} D_{TV}(L_0, L_1) &= |\mathbf{Pr}(S = s_0 \mid \mathcal{E}_0) - \mathbf{Pr}(S = s_0 \mid \mathcal{E}_1)| + \sum_{s\in\Omega\setminus\{s_0\}} |\mathbf{Pr}(S = s \mid \mathcal{E}_0) - \mathbf{Pr}(S = s \mid \mathcal{E}_1)| \\ &\leq \frac{8}{100} + \mathbf{Pr}(S \neq s_0 \mid \mathcal{E}_0) + \mathbf{Pr}(S \neq s_0 \mid \mathcal{E}_1) \leq \frac{24}{100} < \frac{1}{4} \end{aligned} \tag{4.50}$$

which is a contradiction. Thus, we conclude that $q > (k/\varepsilon)/100$, as desired. □

Using the above lemma, we may prove the full hardness of LabelKKMC.

**Lemma 4.15** (Hardness of LabelKKMC)**.** *Suppose an algorithm $\mathcal{A}$, possibly randomized, solves* LabelKKMC *with probability at least* $2/3$ *over the input distribution $\nu_{\mathrm{KKMC}}(k,\varepsilon)$ and the algorithm's random coin tosses. Then, $\mathcal{A}$ makes $\Omega(nk/\varepsilon)$ kernel queries.*

*Proof.* Given such an algorithm, we given an algorithm $\mathcal{B}$ solving LabelSingleKKMC (see definition 4.13) as follows. We first generate $n-1$ points $\{\mathbf{x}_i\}_{i=1}^{n-1}$ drawn i.i.d. from $\nu_{KKMC}$. We then draw a random index $i^* \sim \mathrm{Unif}([n/2])$, set the $i^* + n/2$th point to $\mathbf{x}$, and the rest of the points according to $\{\mathbf{x}_i\}_{i=1}^{n-1}$. We then run $\mathcal{A}$ on this input instance as follows. We can clearly give $\mathcal{A}$ the labels of the first $n/2$ elements of$\{\mathbf{x}_i\}_{i=1}^{n-1}$

without making inner product queries to $\mathbf{x}$. Whenever we need to read a kernel entry that doesn't involve the $i^*$th element, we generate the inner product without making queries to $\mathbf{x}$. Otherwise, we make the required inner product query $\mathbf{x} \cdot \mathbf{v}_{\ell,\ell_1,\ell_2}$ requested by $\mathcal{A}$. With probability at least $2/3$, $\mathcal{A}$ succeeds in outputting $\mathrm{block}(\mathbf{x}_i)$ for at least a $1/6$ fraction of the last $n/2$ input points. By symmetry, $\mathbf{x}$ has a correct label with probability at least $1/6$ in this event. Then by independence, the algorithm succeeds with probability at least $(2/3)(1/6) = 1/9 > 1/100$. Thus, $\mathcal{B}$ indeed solves LabelSingleKKMC.

We now bound the number of inner product queries made by $\mathcal{B}$. Suppose for contradiction that $\mathcal{A}$ makes at most $o(nk/\varepsilon)$ total kernel queries. Then by averaging, $\mathcal{A}$ makes at most $(k/\varepsilon)/200$ kernel queries for a $199/200$ fraction of the $n/2$ last rows. Similarly, $\mathcal{A}$ makes at most $(k/\varepsilon)/200$ kernel queries for a $199/200$ fraction of the last $n/2$ columns. Thus, by a union bound, $\mathcal{A}$ makes at most $(k/\varepsilon)/100$ inner product queries for a $99/100$ fraction of the last $n/2$ input points. By symmetry, $\mathcal{A}$ makes at most $(k/\varepsilon)/100$ inner product queries on $\mathbf{x}$ with probability at least $99/100$. This contradicts Lemma 4.14. Thus, we conclude that $\mathcal{A}$ makes at least $\Omega(nk/\varepsilon)$ kernel queries, as desired. □

Finally, we use the above lemma to show the hardness of kernel $k$-means clustering.

**Corollary 4.16.** *Suppose an algorithm $\mathcal{A}$ gives a $(1+\varepsilon/40)$-approximate kernel $k$-means clustering solution with probability at least $2/3$ over the input distribution $\mathbf{K} \sim \mu_{\mathrm{KKMC}}(n,k,\varepsilon)$ and the algorithm's random coin tosses. Then, $\mathcal{A}$ makes $\Omega(nk/\varepsilon)$ kernel queries.*

*Proof.* Using a $(1+\varepsilon/40)$-approximate algorithm for $k$-means clustering, we can solve the computational problem described above as follows. We first cluster all the points using $\mathcal{A}$. Then, note that by Lemma 4.11, at least $2/5$ of the points must belong to a cluster such that sampling $O(1/\varepsilon)$ points within its cluster allows us to find a point such that at least one coordinate matches a labeled point's coordinate. Then, on average, we will get $1/5$ of these correct and thus $1/6$ of these with very high probability by Chernoff bounds. Note that this used $Q + O(n/\varepsilon)$ kernel queries, where $Q$ is the number of kernel queries that the kernel $k$-means step used. Then, since $Q + O(n/\varepsilon) = \Omega(nk/\varepsilon)$, we have that $Q = \Omega(nk/\varepsilon)$, as desired. □

Finally, we obtain Theorem 4.5 by rescaling $\varepsilon$ by a constant factor.

# 5 Clustering mixtures of Gaussians

In this section, we show that our worst case kernel query complexity lower bounds for the kernel $k$-means clustering problem are pessimistic by a factor of $k$ when our input instance is mixture of $k$ Gaussians. More specifically, we prove the following theorem:

**Theorem 5.1** (Clustering mixtures of Gaussians)**.** *Let $m = \Omega(\varepsilon^{-1}\log n)$ as specified by Corollary 5.3. Suppose we have a mixture of $k$ Gaussians with isotropic covariance $\sigma^2\mathbf{I}_d$ and means $(\boldsymbol{\mu}_\ell)_{\ell=1}^k$ in $\mathbb{R}^d$. Furthermore, suppose that the Gaussian means $\boldsymbol{\mu}_\ell$ are all separated by at least $\left\|\boldsymbol{\mu}_{\ell_1} - \boldsymbol{\mu}_{\ell_2}\right\|_2 \geq \Omega(\sigma\sqrt{\log k})$ as specified by Theorem 5.1 of [RV17] and $\left\|\boldsymbol{\mu}_{\ell_1} - \boldsymbol{\mu}_{\ell_2}\right\|_2 \geq \Omega(\sigma\sqrt{\log\log n + \log\varepsilon^{-1}})$ as specified by Lemma 5.2 with $\delta = (2m+k)^{-3}$. Finally, suppose that we are in the parameter regime of* $\mathrm{poly}(k, 1/\varepsilon, d, \log n) = O(\sqrt{n})$, *$d\varepsilon \geq 1$, and $k/\varepsilon \leq d \leq n/10$. Then, there exists an algorithm outputting a $(1+O(\varepsilon))$-approximate $k$-means clustering solution with probability at least $2/3$.*

## 5.1 Proof overview

By Theorem 5.1 of [RV17], we can in $s = \mathrm{poly}(k, 1/\varepsilon, d)$ samples compute approximations $(\hat{\boldsymbol{\mu}}_\ell)_{\ell=1}^k$ to the true Gaussian means $(\boldsymbol{\mu}_\ell)_{\ell=1}^k$ up to

$$\|\hat{\boldsymbol{\mu}}_\ell - \boldsymbol{\mu}_\ell\|_2^2 \leq \sigma^2 \tag{5.1}$$

by setting $\delta = 1$ in their paper. Set $t \coloneqq \max\{s, 2m+k, d\}$. Then, we may extract the $t$ underlying points in $t^2 = O(n)$ kernel queries by reading a $t \times t$ submatrix of the kernel matrix and retrieving the underlying Gaussian points themselves from the inner product matrix up to a rotation, for instance by Cholesky decomposition. Since we have a sample of size at least $s$, we may approximate the Gaussian means. Now, of the $t$ Gaussian points sampled, we show that we can exactly assign which points belong to which Gaussians for $2m+k$ input points using Lemma 5.2.

Now let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two input points with the same mean. Then note that $\mathbf{x}_1 - \mathbf{x}_2 \sim \mathcal{N}(0, 2\sigma^2 \mathbf{I}_d)$ and that we may compute its inner product with another input point $\mathbf{x}'$ in two kernel queries, i.e. by computing $\mathbf{x}_1 \cdot \mathbf{x}'$ and $\mathbf{x}_2 \cdot \mathbf{x}'$ individually and subtracting them. Now let $\mathbf{S} \in \mathbb{R}^{m \times d}$ be the matrix formed by placing $m$ pairs of the above difference of pairs of Gaussians drawn from the same mean, scaled by $(2\sigma^2)^{-1}$. Then $\mathbf{S}$ is a $m \times d$ matrix of i.i.d. Gaussians, and for $n - 2m$ input points $\mathbf{x}_i$, we may compute $\mathbf{S}\mathbf{x}_i$ with $O(nm)$ kernel queries total. We then prove that for well-separated Gaussian means, $\mathbf{S}\mathbf{x}_i$ can be used to identify which true Gaussian mean $\mathbf{x}_i$ came from in corollary 5.3.

Finally, we show that clustering points to their Gaussian means results in an approximately optimal $k$-means clustering. By the above, we can do this assigning for Gaussian means that are separated by more than $\varepsilon\sigma^2 d$, and otherwise, assigning to a wrong mean only $\varepsilon\sigma^2 d$ away still results in a $(1+\varepsilon)$-optimal clustering.

## 5.2 Assigning input points to Gaussian means

We first present the following lemma, which allows us to distinguish whether a point is drawn from a Gaussian with one mean or another with probability at least $1-\delta$.

**Lemma 5.2** (Distinguishing Gaussian means)**.** *Let $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$ be two Gaussian means separated by $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2 \geq C\sigma^2 \log \delta^{-1}$ for a constant $C$ large enough and $\delta \in (0, 1/2)$. Furthermore, let $\hat{\boldsymbol{\theta}}_1, \hat{\boldsymbol{\theta}}_2$ be approximations to the Gaussian means with $\left\|\hat{\boldsymbol{\theta}}_b - \boldsymbol{\theta}_b\right\|_2 \leq \sigma$ for $b \in \{1, 2\}$. Let $\hat{\mathbf{c}} := (\hat{\boldsymbol{\theta}}_1 + \hat{\boldsymbol{\theta}}_2)/2$. Then*

$$\begin{cases} (\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) > 0 & \textit{if } \mathbf{x} \sim \mathcal{N}(\boldsymbol{\theta}_1, \sigma^2 \mathbf{I}_d) \textit{ w.p. at least } 1-\delta \\ (\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) < 0 & \textit{if } \mathbf{x} \sim \mathcal{N}(\boldsymbol{\theta}_2, \sigma^2 \mathbf{I}_d) \textit{ w.p. at least } 1-\delta \end{cases}. \tag{5.2}$$

*Proof.* Let $\mathbf{x} = \boldsymbol{\theta} + \boldsymbol{\eta}$ with $\boldsymbol{\theta} \in \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ and $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$. Then if $\boldsymbol{\theta} = \boldsymbol{\theta}_1$, then

$$\begin{aligned} (\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) &= (\boldsymbol{\theta}_1 - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) + \boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \\ &= \left\|\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}\right\|_2^2 + (\boldsymbol{\theta}_1 - \hat{\boldsymbol{\theta}}_1) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) + \boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \end{aligned} \tag{5.3}$$

and similarly if $\boldsymbol{\theta} = \boldsymbol{\theta}_2$, then

$$\begin{aligned} (\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) &= (\boldsymbol{\theta}_2 - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) + \boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \\ &= (\hat{\boldsymbol{\theta}}_2 - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) + (\boldsymbol{\theta}_2 - \hat{\boldsymbol{\theta}}_2) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) + \boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}). \end{aligned} \tag{5.4}$$

Note that

$$\left\|\frac{\hat{\boldsymbol{\theta}}_1 - \hat{\boldsymbol{\theta}}_2}{2} - \frac{\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2}{2}\right\|_2 \leq \frac{1}{2}\left(\left\|\boldsymbol{\theta}_1 - \hat{\boldsymbol{\theta}}_1\right\|_2 + \left\|\boldsymbol{\theta}_2 - \hat{\boldsymbol{\theta}}_2\right\|_2\right) \leq \sigma \tag{5.5}$$

and thus we have the following estimates:

$$\begin{aligned} \left\|\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}\right\|_2^2 &\geq \|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 - 2\sigma\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 \\ (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_2 - \hat{\mathbf{c}}) &\leq -\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 + \sigma^2 + 2\sigma\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 \\ \left|(\boldsymbol{\theta}_1 - \hat{\boldsymbol{\theta}}_1) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}})\right| &\leq \sigma^2 + \sigma\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 \\ \left|(\boldsymbol{\theta}_2 - \hat{\boldsymbol{\theta}}_2) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}})\right| &\leq \sigma^2 + \sigma\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2. \end{aligned} \tag{5.6}$$

These bound all but the last terms in equations (5.3) and (5.4). To bound the last term, note also that we may take $\sigma \leq \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2/12$ for $C \geq 144/\log 2$. Furthermore, $\boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \sim \mathcal{N}(0, \sigma^2 \left\|\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}\right\|_2^2)$ and thus with probability at least $1-\delta$, we have that

$$\left|\boldsymbol{\eta} \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}})\right| \leq \sigma\sqrt{\log\frac{1}{\delta}}\left\|\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}\right\|_2 \leq \frac{1}{6}\|\boldsymbol{\theta}_1 - \mathbf{c}\|(\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 + \sigma) \leq \frac{1}{3}\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 \tag{5.7}$$

by taking $C \geq 6$. We also have the bounds

$$\begin{aligned} \sigma^2 &\leq \frac{\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2}{144} \\ 2\sigma\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 &\leq 2\sigma\frac{\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2}{6}\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2 \leq \frac{1}{3}\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2. \end{aligned} \tag{5.8}$$

Then, with probability at least $1-\delta$, if $\boldsymbol{\theta} = \boldsymbol{\theta}_1$, then

$$(\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \geq \|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 - \frac{1}{3}\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 > 0 \tag{5.9}$$

and if $\boldsymbol{\theta} = \boldsymbol{\theta}_2$, then

$$(\mathbf{x} - \hat{\mathbf{c}}) \cdot (\hat{\boldsymbol{\theta}}_1 - \hat{\mathbf{c}}) \leq -\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 + \frac{\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2}{36} + \frac{1}{3}\|\boldsymbol{\theta}_1 - \mathbf{c}\|_2^2 < 0 \tag{5.10}$$

and thus we conclude as desired. □

Using Lemma 5.2, we may identify the true Gaussian mean of a point with probability at least $1-(2m+k)^{-3}$ with squared separation only $O(\sigma^2(\log\log n + \log\varepsilon^{-1} + \log k))$. Then by a union bound, we may indeed identify the true Gaussian means of $2m+k$ points simultaneously with high probability. We may thus form the matrix $\mathbf{S}$ of i.i.d. standard Gaussians as described previously and apply it to the $n-2m$ remaining points.

As a corollary of Lemma 5.2, we show that for Gaussian means that are separated more, with squared distance at least $\varepsilon\sigma^2 d$, we may distinguish the means with a Gaussian sketch of dimension $m = O(\varepsilon^{-1}\log\delta^{-1})$ with probability at least $1-\delta$. In particular, we may choose the failure probability to be $\delta = (nk)^{-3}$ so that with a sketch dimension of $m = O(\varepsilon^{-1}\log(nk)^3) = O(\varepsilon^{-1}\log n)$, we can identify the correct Gaussian mean for all $n-2m$ remaining input points simultaneously by the union bound, as claimed. That is, using $\mathbf{Sx}$, we can find the correct mean of $\mathbf{x}$ for Gaussians with large enough separation.

**Corollary 5.3.** *Let $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \in \mathbb{R}^d$ be two Gaussian means separated by $\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 \geq \varepsilon\sigma^2 d$, and let $\delta \in (0, 1/2)$. Let $\mathbf{S} \in \mathbb{R}^{m\times d}$ be a matrix of i.i.d. standard Gaussians. If $m \geq C\varepsilon^{-1}\log(\delta^{-1})$, for some constant $C$ large enough, then there exists an algorithm that can decide whether $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_1, \sigma^2\mathbf{I}_d)$ or $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_2, \sigma^2\mathbf{I}_d)$ given only $\mathbf{S}$, $\mathbf{Sx}$, and the approximate means $\hat{\boldsymbol{\mu}}_j$, with probability at least $1-\delta$.*

*Proof.* Let $\mathbf{S} = \mathbf{U\Sigma V}^\top$ be the truncated SVD of $\mathbf{S}^\top$. Note that the algorithm can compute this decomposition and thus can retrieve $\mathbf{V}^\top\mathbf{x} = (\mathbf{U\Sigma})^{-1}\mathbf{Sx}$ and that $\mathbf{V}^\top$ is a random projection. Then as discussed in the proof of Theorem 2.1 in [DG03], we have

$$\mathop{\mathbf{E}}_{\mathbf{S}}\left(\left\|\mathbf{V}^\top(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)\right\|_2^2\right) = \frac{m}{d}\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2, \tag{5.11}$$

and by Lemma 2.2 of [DG03],

$$\mathop{\mathbf{Pr}}_{\mathbf{S}}\left(\left\|\mathbf{V}^\top(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)\right\|_2^2 \leq \frac{1}{2}\frac{m}{d}\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2\right) < \exp(-\Omega(m)) \leq \frac{\delta}{2} \tag{5.12}$$

for $C$ chosen large enough. Now suppose that the above event does not happen, which happens with probability at least $1-\delta/2$. Write $\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\eta}$, where $\boldsymbol{\mu} \in \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2\}$ and $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$. Note then that $\mathbf{V}^\top\mathbf{x} = \mathbf{V}^\top\boldsymbol{\mu} + \mathbf{V}^\top\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{V}^\top\boldsymbol{\mu}, \sigma^2\mathbf{I}_m)$ by the rotational invariance of Gaussians, and

$$\left\|\mathbf{V}^\top\boldsymbol{\mu}_1 - \mathbf{V}^\top\boldsymbol{\mu}_2\right\|_2^2 \geq \frac{1}{2}\frac{m}{d}\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 \geq \frac{1}{2}\frac{m}{d}\varepsilon\sigma^2 d \geq \frac{C}{2}\log\delta^{-1}. \tag{5.13}$$

Furthermore, we have an approximation of the means $\mathbf{V}^\top\boldsymbol{\mu}_1$ and $\mathbf{V}^\top\boldsymbol{\mu}_2$ via $\mathbf{V}^\top\hat{\boldsymbol{\mu}}_1$ and $\mathbf{V}^\top\hat{\boldsymbol{\mu}}_2$ with

$$\left\|\mathbf{V}^\top(\boldsymbol{\mu}_b - \hat{\boldsymbol{\mu}}_b)\right\|_2 \leq \|\boldsymbol{\mu}_b - \hat{\boldsymbol{\mu}}_b\|_2 = \sigma^2. \tag{5.14}$$

We then take our $C$ here to be big enough to use Lemma 5.2 and conclude. □

We now put corollary 5.3 to algorithmic use by using it to assign to each point a center withing $\varepsilon\sigma^2 d$.

**Lemma 5.4.** *With probability at least* 99/100*, we may simultaneously assign for each* $\mathbf{x}_i$ *for* $i \in [n]$ *a center* $\boldsymbol{\mu}_{\ell_i}$ *with* $\left\|\boldsymbol{\mu}_{\ell_i} - \boldsymbol{\mu}_{\ell_i^*}\right\|_2^2 \leq \varepsilon\sigma^2 d$*, where* $\boldsymbol{\mu}_{\ell_i^*}$ *is the true Gaussian mean that generated* $\mathbf{x}_i$*. Furthermore, the assignment algorithm that we describe only depends on* $\mathbf{S}$*,* $\mathbf{S}\mathbf{x}_i$*, and approximate means* $\hat{\boldsymbol{\mu}}_j$*.*

*Proof.* We claim that we can assign a center within squared distance $\varepsilon\sigma^2 d$ as follows. Let $\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\eta}$ with $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$. We then iterate through guesses $\boldsymbol{\mu}_j$ for $j \in [k]$ and assign $\boldsymbol{\mu}_j$ to $\mathbf{x}$ if we run the hypothesis test between $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_j, \sigma^2\mathbf{I}_d)$ and $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \sigma^2\mathbf{I}_d)$ and $\mathbf{x}$ chooses $\boldsymbol{\mu}_j$ for *every* $\ell \in [k] \setminus \{j\}$. Recall that we set the failure rate $\delta$ in corollary 5.3 to $(nk)^{-3}$, so the hypothesis test is accurate for all $nk(k-1)$ hypothesis tests ranging over $n$ data points, $j \in [k]$, and $\ell \in [k] \setminus \{\ell\}$. Clearly, guessing $\boldsymbol{\mu}_j = \boldsymbol{\mu}$ results in passing all the hypothesis tests in this case. Note that our hypothesis test is run using corollary 5.3 and only depends on $\mathbf{S}$, $\mathbf{S}\mathbf{x}_i$, and approximate means $\hat{\boldsymbol{\mu}}_j$.

Now suppose that $\boldsymbol{\mu}_j$ is a center that is more than $\varepsilon\sigma^2 d$ squared distance away from $\boldsymbol{\mu}$. Then when we guess $\boldsymbol{\mu}_j$, $\boldsymbol{\mu}_j$ fails at least one hypothesis test, namely the one testing $\mathcal{N}(\boldsymbol{\mu}_j, \sigma^2\mathbf{I}_d)$ against $\mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I}_d)$ when $\boldsymbol{\mu}_\ell = \boldsymbol{\mu}$. Thus, this algorithm correctly assigns every Gaussian input point to a center that is at most $\varepsilon\sigma^2 d$ square distance away from the true mean $\boldsymbol{\mu}$. □

## 5.3 Clustering the points

Now that we have approximately assigned input points to Gaussian means in $O(nm) = \tilde{O}(n/\varepsilon)$ kernel queries, it remains to show that this information suffices to give a $(1+\varepsilon)$-approximate solution to the KKMC problem.

**Theorem 5.5.** *Let* $d\varepsilon \geq 1$ *and* $k/\varepsilon \leq d \leq n/10$ *and let our data set* $\{\mathbf{x}_i\}_{i=1}^n$ *be distributed as a mixture of* $k$ *Gaussians as described before. Then assigning the* $\mathbf{x}_i$ *to approximate means as in Lemma 5.4 gives a* $(1+8\varepsilon)$*-approximate* $k$*-means clustering solution with probability at least* 98/100*.*

*Proof.* Let $\mathbf{X} \in \mathbb{R}^{n\times d}$ be the design matrix of points with our dataset drawn from a mixture of Gaussians in the rows. Now write $\mathbf{X} = \mathbf{M} + \mathbf{G}$, where $\mathbf{M}$ is the matrix with the Gaussian mean of each point in the rows and $\mathbf{G}$ is a matrix with rows all distributed as $\mathcal{N}(0, \sigma^2\mathbf{I}_d)$.

**Lower bounds on the cost.** We first bound below the cost of any $k$-means clustering solution, i.e. an assignment of $k$ centers to each of the $n$ points. We may then place these centers in the rows of a matrix $\mathbf{C}$, so that the input data point $\mathbf{x}_i = \mathbf{e}_i^\top \mathbf{X}$ is assigned the $k$-means center $\mathbf{e}_i^\top \mathbf{C}$ for $i \in [n]$. The cost of this $k$-means solution is then

$$\|\mathbf{X} - \mathbf{C}\|_F^2 = \|\mathbf{G} + \mathbf{M} - \mathbf{C}\|_F^2. \tag{5.15}$$

Now note that $\mathbf{M} - \mathbf{C}$ has rank at most $2k$, so the above cost is bounded below by the cost of the best rank $2k$ approximation of $\mathbf{G}$ in Frobenius norm. Furthermore, by the Eckart-Young-Mirky theorem, the cost of the optimal low rank approximation is the sum of the smallest $d - 2k$ squared singular values of $\mathbf{G}$.

Let $s_1(\mathbf{G}) \geq s_2(\mathbf{G}) \geq \cdots \geq s_d(\mathbf{G})$ denote the singular values of $\mathbf{G}$. Note that $\mathbf{G}/\sigma$ is a matrix with i.i.d. standard Gaussians, so by results summarized in [RV09], we have that $(1/2)\sigma\sqrt{n} \leq s_n(\mathbf{G}) \leq s_1(\mathbf{G}) \leq (3/2)\sigma\sqrt{n}$ with probability 99/100 for $n$ large enough and $d \leq n/10$. Then,

$$\sum_{i=1}^{2k} s_i(\mathbf{G})^2 \leq (2k)s_1(\mathbf{G})^2 \leq 3(2k)s_n(\mathbf{G})^2 \leq \frac{6k}{d}\sum_{i=1}^{d} s_i(\mathbf{G})^2 \tag{5.16}$$

and thus

$$\|\mathbf{X} - \mathbf{C}\|_F^2 \geq \sum_{i=1}^{d-2k} s_i(\mathbf{G})^2 \geq \left(1 - \frac{6k}{d}\right)\sum_{i=1}^{d} s_i(\mathbf{G})^2 = \left(1 - \frac{6k}{d}\right)\|\mathbf{G}\|_F^2 \geq (1-6\varepsilon)\|\mathbf{G}\|_F^2. \tag{5.17}$$

**The cost of clustering by the Gaussian means.** We now give an algorithm using the approximate Gaussian means and our Gaussian mean assignment algorithms. Note that if we can correctly cluster every input point to its Gaussian center, then the resulting clustering has cost at most $\|\mathbf{G}\|_F^2$ since using the empirical center of the Gaussians will have lower cost than the true means. Then for $d \geq k/\varepsilon$, we have that this clustering has cost at most $1/(1-4\varepsilon) \leq 1+5\varepsilon$ times the optimal clustering by equation (5.17).

However, with our kernel query budget, we can only do this assignment for Gaussian means that are separated by squared distance $\varepsilon\sigma^2 d$ using Lemma 5.2; for separation smaller than this, we cannot disambiguate. The fix here is that we in fact do not need to, since assigning to a Gaussian mean that is less than $\varepsilon\sigma^2 d$ does not change the cost by more than a $(1+\varepsilon)$ for that point.

Now consider an input point $\mathbf{x} = \boldsymbol{\mu}^* + \boldsymbol{\eta}$ with $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$, let $\boldsymbol{\mu}$ be the true mean assigned to $\mathbf{x}$ in Lemma 5.4, and let $\hat{\boldsymbol{\mu}}$ be the approximation that approximates $\boldsymbol{\mu}$. We then have that

$$\|\mathbf{x} - \hat{\boldsymbol{\mu}}\|_2^2 = \|\boldsymbol{\eta} + (\boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}})\|_2^2 = \|\boldsymbol{\eta}\|_2^2 + \|\boldsymbol{\mu}_* - \hat{\boldsymbol{\mu}}\|_2^2 + 2\langle \boldsymbol{\eta}, \boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\rangle. \tag{5.18}$$

Note that $\|\boldsymbol{\mu}_* - \hat{\boldsymbol{\mu}}\|_2^2 \leq \|(\boldsymbol{\mu}_* - \boldsymbol{\mu}) + (\boldsymbol{\mu} - \hat{\boldsymbol{\mu}})\|_2^2 \leq 2\varepsilon\sigma^2 d + 2\sigma^2$ and that $\langle \boldsymbol{\eta}, \boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\rangle \sim \mathcal{N}(0, \sigma^2 \|\boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\|_2^2)$. Thus, we have that

$$\|\mathbf{x} - \hat{\boldsymbol{\mu}}\|_2^2 = \|\boldsymbol{\eta}\|_2^2 + 2\varepsilon\sigma^2 d + 2\sigma^2 + 2\langle \boldsymbol{\eta}, \boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\rangle \leq \|\boldsymbol{\eta}\|_2^2 + 4\varepsilon\sigma^2 d + 2\langle \boldsymbol{\eta}, \boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\rangle. \tag{5.19}$$

Note that summing the $2\langle \boldsymbol{\eta}, \boldsymbol{\mu}^* - \hat{\boldsymbol{\mu}}\rangle$ term over $n$ input points gives a zero mean Gaussian with variance at most $2n\sigma^2(2\varepsilon\sigma^2 d + 2\sigma^2) \leq 4n\sigma^4\varepsilon d$. With probability 99/100, this is bounded by $4\sigma^2\sqrt{n\varepsilon d} \leq \varepsilon n\sigma^2 d$ for $n$ large enough.

Thus, we then have that

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \hat{\boldsymbol{\mu}}\|_2^2 \leq \|\mathbf{G}\|_F^2 + 5\varepsilon(n\sigma^2 d). \tag{5.20}$$

Now note that $\|\mathbf{G}\|_F^2/\sigma^2$ is a chi-squared variable with $nd$ degrees of freedom, so by concentration bounds found in [LM00], we have that

$$\mathbf{Pr}\Big(\|\mathbf{G}\|_F^2/\sigma^2 - nd \leq 2nd\Big) \leq \exp(-nd) \tag{5.21}$$

and thus with probability at least 99/100 for $nd$ large enough,

$$5\varepsilon(n\sigma^2 d) \leq \frac{5}{3}\varepsilon\|\mathbf{G}\|_F^2. \tag{5.22}$$

We thus conclude that with probability at least 98/100, the above algorithm gives an approximation ratio of at most

$$\frac{1 + 5\varepsilon/3}{1 - 6\varepsilon} \leq 1 + 8\varepsilon \tag{5.23}$$

as claimed. □

# 6 Acknowledgements

We would like to thank the anonymous reviewers for helpful feedback. D. Woodruff would like to thank support from the Office of Naval Research (ONR) grant N00014-18-1-2562. This work was also partly done while D. Woodruff was visiting the Simons Institute for the Theory of Computing.

# References

[ACW17] Haim Avron, Kenneth L Clarkson, and David P Woodruff. Sharper bounds for regularized data fitting. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*, pages 27:1–27:22, 2017.

[ADHP09] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

[Bac13] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, pages 185–209, 2013.

[BDM09] Christos Boutsidis, Petros Drineas, and Michael W Mahoney. Unsupervised feature selection for the $k$-means clustering problem. In *Advances in Neural Information Processing Systems*, pages 153–161, 2009.

[BIS17] Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In *Advances in Neural Information Processing Systems*, pages 4308–4318, 2017.

[BJ02] Francis R Bach and Michael I Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3(Jul):1–48, 2002.

[BYP02] Ziv Bar-Yossef and Christos H Papadimitriou. *The complexity of massive data set computations.* PhD thesis, University of California, Berkeley, 2002.

[CBMS15] Nicolo Cesa-Bianchi, Yishay Mansour, and Ohad Shamir. On the complexity of learning with kernels. In *Conference on Learning Theory*, pages 297–325, 2015.

[CEM$^+$15] Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for $k$-means clustering and low rank approximation. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, pages 163–172. ACM, 2015.

[DG03] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Struct. Algorithms*, 22(1):60–65, 2003.

[EAM15] Ahmed El Alaoui and Micahel W Mahoney. Fast randomized kernel methods with statistical guarantees. In *Advances in Neural Information Processing Systems*, pages 775–783, 2015.

[FHT01] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 2001.

[FL11] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, pages 569–578. ACM, 2011.

[FS01] Shai Fine and Katya Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2(Dec):243–264, 2001.

[Har75] John A Hartigan. *Clustering algorithms.* Wiley, 1975.

[LM00] Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pages 1302–1338, 2000.

[LWZ14] Ming Lin, Shifeng Weng, and Changshui Zhang. On the sample complexity of random Fourier features for online learning: How many random Fourier features do we need? *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(3):13, 2014.

[MM17] Cameron Musco and Christopher Musco. Recursive sampling for the Nyström method. In *Advances in Neural Information Processing Systems*, pages 3833–3845, 2017.

[MS17] Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5788–5799, 2017.

[MW17] Cameron Musco and David P Woodruff. Sublinear time low-rank approximation of positive semidefinite matrices. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science*, pages 672–683. IEEE, 2017.

[RR08] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2008.

[RV09] Mark Rudelson and Roman Vershynin. Smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 62(12):1707–1739, 2009.

[RV17] Oded Regev and Aravindan Vijayaraghavan. On learning mixtures of well-separated Gaussians. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science*, pages 85–96. IEEE, 2017.

[SS01] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.

[STV+04] Bernhard Schölkopf, Koji Tsuda, Jean-Philippe Vert, Director Sorin Istrail, Pavel A Pevzner, Michael S Waterman, et al. *Kernel methods in computational biology.* MIT press, 2004.

[TLB+11] Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Kalai. Adaptively learning the crowd kernel. In *Proceedings of the 28th International Conference on Machine Learning*, pages 673–680. ACM, 2011.

[Yao77] Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual IEEE Symposium on Foundations of Computer Science*, pages 222–227. IEEE, 1977.

[YPW17] Yun Yang, Mert Pilanci, and Martin J Wainwright. Randomized sketches for kernels: Fast and optimal nonparametric regression. *The Annals of Statistics*, 45(3):991–1023, 2017.

[Zha05] Tong Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17(9):2077–2098, 2005.

[ZMLS07] Jianguo Zhang, Marcin Marszałek, Svetlana Lazebnik, and Cordelia Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2007.