Sparse Approximation of a Kernel Mean

Efrén Cruz Cortés and Clayton Scott, *Member, IEEE*

***Abstract*—Kernel means are frequently used to represent probability distributions in machine learning problems. In particular, the well known kernel density estimator and the kernel mean embedding both have the form of a kernel mean. Unfortunately, kernel means face scalability issues. A single point evaluation of the kernel density estimator, for example, requires a computation time linear in the training sample size. To address this challenge, we present a method to efficiently construct a sparse approximation of a kernel mean. We do so by first establishing an incoherence-based bound on the approximation error. We then observe that, for any kernel with constant norm (which includes all translation invariant kernels), the bound can be efficiently minimized by solving the *k*-center problem. The outcome is a linear time construction of a sparse kernel mean, which also lends itself naturally to an automatic sparsity selection scheme. We demonstrate the computational gains of our method by looking at several benchmark datasets, as well as three applications involving kernel means: Euclidean embedding of distributions, class proportion estimation, and clustering using the mean-shift algorithm.**

***Index Terms*—Sparse approximation, *k*-center problem, kernel density estimator, kernel mean embedding.**

I. Introduction

A KERNEL *mean* is a quantity of the form

$$\frac{1}{n}\sum_{i=1}^{n}\phi(\cdot, x_i), \tag{1}$$

where ϕ is a *kernel* and $x_1, \ldots, x_n \in \mathbb{R}^d$ are data points. We define kernels rigorously below. Our treatment includes many common examples of kernels, such as the Gaussian kernel, and encompasses both symmetric positive definite kernels and kernels used for nonparametric density estimation.

Kernel means arise frequently in machine learning and nonparametric statistics as representations of probability distributions. In this context, $x_1, \ldots, x_n$ are understood to be realizations of some unknown probability distribution. The kernel density estimator (KDE) is a kernel mean that estimates the density of the data. The kernel mean embedding (KME) is a kernel mean that maps the probability distribution into a reproducing kernel Hilbert space. These two motivating applications of kernel means are reviewed in more detail below.

This work is concerned with efficient computation of a sparse approximation of a kernel mean, taking the form

$$\sum_{i=1}^{n}\alpha_i\phi(\cdot, x_i) \tag{2}$$

where $\alpha_i \in \mathbb{R}$ and $k := |\{i : \alpha_i \neq 0\}| \ll n$. In other words, given $x_1, \ldots, x_n$, a kernel ϕ, and a target sparsity k, we seek a sparse kernel mean (2) that accurately approximates the kernel mean (1). This problem is motivated by applications where n is so large that evaluation or manipulation of the full kernel mean is computationally prohibitive. A sparse kernel mean can be evaluated or manipulated much more efficiently. In the large n regime, the sparse approximation algorithm itself must be scalable, and as we argue below, existing sparse approximation strategies are too slow.

Our primary contribution is an efficient algorithm for sparsely approximating a kernel mean. The algorithm results from minimizing a sparse approximation bound based on a novel notion of incoherence. We show that for a broad class of kernels minimizing the sparse approximation bound is equivalent to solving the k-center problem on $x_1, \ldots, x_n$, which in turn leads to an efficient algorithm. An advantage of our approach is that it approximates an arbitrary kernel mean, so we need not address the KDE and KME problems independently, but through a shared methodology.

The rest of the paper is outlined as follows. In Section II we review the KDE and KME, which motivate this work, and also introduce a general definition of kernel that encompasses both of these settings. We review related work and its connection to our contributions in Section III. In Section IV we establish an incoherence-based sparse approximation bound. We then use the principle of bound minimization in Section V to derive a scalable algorithm for sparse approximation of kernel means, and present a sparsity auto-selection scheme. Finally, to demonstrate the efficacy of our approach, Section VI applies our methodology in three different machine learning problems that rely on large-scale KDEs and KMEs, and also presents its performance on 11 benchmark datasets. A preliminary version of this work appeared in [1]. A Matlab implementation of our algorithm is available at [2].

II. Motivation and Formal Setting

Our work is motivated by two primary examples of kernel means. We review the KDE and KME separately, and then pro-

Manuscript received January 6, 2016; revised June 15, 2016 and August 2, 2016; accepted August 5, 2016. Date of publication November 14, 2016; date of current version January 6, 2017. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Masahiro Yukawa. This work was supported in part by the NSF Awards 0953135, 1047871, 1217880, and 1422157.

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: encc@umich.edu; clayscot@umich.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes an appendix with additional experiments and a theoretical result. The file is 663 KB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2016.2628353

pose a general notion of kernel that encompasses the essential features of both settings and is sufficient for addressing the sparse approximation problem. By way of notation, we denote $[n] := \{1, \ldots, n\}$.

A. Kernel Density Estimation

Let $\{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ be a random sample from a distribution with density f. In the context of kernel density estimation, a kernel is a function ϕ such that for all x', $\int \phi(x, x')dx = 1$. In addition, ϕ is sometimes also chosen to be nonnegative, although this is not necessary for theoretical properties such as consistency. The kernel density estimator of f is the function

$$\widehat{f} = \frac{1}{n} \sum_{i \in [n]} \phi(\cdot, x_i).$$

The KDE is used as an ingredient in a number of machine learning methodologies. For example, a common approach to classification is a plug-in rule that estimates the class-conditional densities with separate KDEs [3], [4]. In anomaly detection, a detector of the form $\widehat{f}(x) \gtrless \gamma$ is commonly employed to determine if a new realization comes from f [5], [6], [7], [8]. In clustering, the mean-shift algorithm forms a KDE and associates each data point to the mode of the KDE that is reached by hill-climbing [9].

Evaluating the KDE at a single test point requires $O(n)$ kernel evaluations, which is undesirable and perhaps prohibitive for large n. On the other hand, a sparse approximation with sparsity k requires only $O(k)$ kernel evaluations. This problem is magnified in algorithms such as mean-shift, where a (derivative of a) KDE is evaluated numerous times for each data point. In our experiments below, we demonstrate the computational savings of our approach in KDE-based algorithms for the embedding of probability distributions and mean-shift clustering.

B. Kernel Mean Embedding of Distributions

Let $\{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ be a random sample from a distribution P. A *symmetric positive definite kernel* is a function $\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that is symmetric and is such that all square matrices of the form $[\phi(x_i, x_j)]_{i,j=1}^n$ are positive semidefinite. Every symmetric positive definite kernel is associated to a unique Hilbert space of functions called a reproducing kernel Hilbert space (RKHS), which can be thought of as the closed linear span of $\{\phi(\cdot, x) \mid x \in \mathbb{R}^d\}$ [10]. The RKHS has a property known as the *reproducing property* which states that for all f in the RKHS, $f(x) = \langle f, \phi(\cdot, x)\rangle$, where $\langle \cdot, \cdot \rangle$ denotes the inner product.

The idea behind the kernel mean embedding is to select a symmetric positive definite kernel ϕ, and embed P in the RKHS associated with ϕ via the mapping

$$\Psi(P) := \int \phi(\cdot, x) dP(x).$$

Since P is unknown, this mapping is estimated via the kernel mean

$$\widehat{\Psi}(P) := \frac{1}{n} \sum_{i \in [n]} \phi(\cdot, x_i).$$

The utility of the KME derives from the fact that for certain kernels, Ψ is injective. This permits the treatment of probability distributions as objects in a Hilbert space, which allows many existing machine learning methods to be applied in problems where probability distributions play the role of feature vectors [11], [12], [13], [14]. For example, suppose that random samples of size n are available from several probability distributions $P_1, \ldots, P_N$. A KME-based algorithm will require the computation of all pairs of inner products of kernel mean embeddings of these distributions. If $x_1, \ldots, x_n \sim P$ and $x'_1, \ldots, x'_n \sim P'$, then $\langle \widehat{\Psi}(P), \widehat{\Psi}(P')\rangle = \frac{1}{n^2} \sum_{i,j} \phi(x_i, x'_j)$ by the reproducing property. Therefore the calculation of all pairwise inner products of kernel mean embeddings requires $O(N^2 n^2)$ kernel evaluations. On the other hand, if we have sparse representations of the kernel means, these pairwise inner products can be calculated with only $O(N^2 k^2)$ kernel evaluations, a substantial computational savings. In our experiments below, we demonstrate the computational savings of our approach in KME-based algorithms for the embedding of probability distributions and class-proportion estimation.

C. Generalized Notion of Kernel

The problem of sparsely approximating a sample mean can be addressed more generally in an inner product space. This motivates the following definition of kernel, which is satisfied by both density estimation kernels and symmetric positive definite kernels.

Definition 1: We say that $\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *kernel* if there exists an inner product space $\mathcal{H}$ such that for all x in $\mathbb{R}^d$, $\phi(\cdot, x) \in \mathcal{H}$.

In the case of kernel density estimation, all commonly used kernels satisfy $\phi(\cdot, x) \in L^2(\mathbb{R}^d)$ for all $x \in \mathbb{R}^d$. Here, $L^2(\mathbb{R}^d)$ is the space of equivalence classes of square integrable functions. When we write $\phi(\cdot, x) \in L^2(\mathbb{R}^d)$, we view $\phi(\cdot, x)$ as a representative of its equivalence class. In the case of the kernel mean embedding, we may simply take $\mathcal{H}$ to be the RKHS associated with ϕ.

Definition 2: For $\{x_1, \ldots, x_n\}$ a subset of $\mathbb{R}^d$ and ϕ a kernel with associated inner product space $\mathcal{H}$, we call $K := (\langle \phi(\cdot, x_i), \phi(\cdot, x_j)\rangle_{\mathcal{H}})_{i,j \in [n]}$ the *kernel matrix*.

Our proposed methodology applies to translation invariant kernels and beyond. For concreteness, however, we focus on radial kernels because of the connection to Euclidean geometry.

Definition 3: We say $\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *radial kernel* if ϕ is a kernel as in Def. 1 and there exists a strictly decreasing function $g : [0, \infty) \to \mathbb{R}$ such that, for all $x, x' \in \mathbb{R}^d$,

$$\langle \phi(\cdot, x), \phi(\cdot, x')\rangle_{\mathcal{H}} = g(\|x - x'\|_2).$$

We now review some common examples of radial kernels. The Gaussian kernel with parameter $\sigma > 0$ has the form

$$\phi(x, x') = c_\sigma \exp\left(-\frac{\|x - x'\|_2^2}{2\sigma^2}\right),$$

the Laplacian kernel with parameter $\gamma > 0$ has the form

$$\phi(x, x') = c_\gamma \exp\left(-\frac{\|x - x'\|_2}{\gamma}\right),$$

and the Student-type kernel with parameters $\alpha, \beta > 0$ has the form

$$\phi(x, x') = c_{\alpha,\beta}\left(1 + \frac{\|x - x'\|_2^2}{\beta}\right)^{-\alpha}.$$

The parameters c_σ, c_γ and $c_{\alpha,\beta}$ can be set to 1 for the KME, or so as to normalize ϕ to be a density estimation kernel, depending on the application.

These examples illustrate that the space $\mathcal{H}$ such that $\phi(\cdot, x) \in \mathcal{H}$ is not unique. Indeed, each of these three kernels is a symmetric positive definite kernel, and therefore we may take $\mathcal{H}$ to be the RKHS associated with ϕ [10], [15]. On the other hand, we may also select $\mathcal{H} = L^2(\mathbb{R}^d)$.

Each of these three examples is also a radial kernel. If we take $\mathcal{H}$ to be the RKHS, then by the reproducing property we simply have $\langle \phi(\cdot, x), \phi(\cdot, x') \rangle = \phi(x, x')$, and in each case, $\phi(x, x') = g(\|x - x'\|)$ for some strictly decreasing g. These kernels are also radial if we take $\mathcal{H} = L^2(\mathbb{R}^d)$. For example, consider the Gaussian kernel, and let us write $\phi = \phi_\sigma$ to indicate the dependence on the bandwidth parameter. Then $\langle \phi_\sigma(\cdot, x), \phi_\sigma(\cdot, x') \rangle_{L^2} = \phi_{\sqrt{2}\sigma}(x, x')$. Similarly, for the Student kernel with $\alpha = (1 + d)/2$ (the Cauchy kernel), we have $\langle \phi_\beta(\cdot, x), \phi_\beta(\cdot, x') \rangle_{L^2} = \phi_{2\beta}(x, x')$. For other kernels, although there may not be a closed form expression for g, it can still be argued that such a g exists, which is all we will need.

D. Abstract Problem Formulation

In the interest of generality and clarity, we consider the problem of sparsely approximating a sample mean in a more abstract setting. Thus, let $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ be an inner product space with induced norm $\| \cdot \|_{\mathcal{H}}$, and let $\{z_1, \ldots, z_n\} \subset \mathcal{H}$. For $\alpha \in \mathbb{R}^n$, define $\|\alpha\|_0 := |\{i \mid \alpha_i \neq 0\}|$. Given an integer $k \leq n$, our objective is to approximate the sample mean $\bar{z} = \frac{1}{n}\sum_i z_i$ as a k-sparse linear combination of $z_1, \ldots, z_n$. In particular, we want to solve the problem

$$\begin{aligned} &\text{minimize} \ \ \|\bar{z} - z_\alpha\|_{\mathcal{H}} \\ &\text{subject to} \ \ \|\alpha\|_0 = k \end{aligned} \tag{3}$$

where $z_\alpha = \sum_{i \in [n]} \alpha_i z_i$.

Note that problem (3) is of the form of the standard sparse approximation problem [16], where $\{z_1, \ldots, z_n\}$ is the so-called *dictionary* out of which the sparse approximation is built. Later we argue that existing sparse approximation algorithms are not suitable from a scalability perspective. Instead, we develop an approach that leverages the fact that the vector being sparsely approximated is the sample mean of the dictionary elements. We are most interested in the case where $z_i = \phi(\cdot, x_i)$ and ϕ is a kernel, but the discussion in Section IV is held in this more abstract sense.

III. Related Work and Contributions

Problem (3) is a specific case of the sparse approximation problem. Since in general it is NP-hard many efforts have been made to approximate its solution in a feasible amount of time. See [16] for an overview. A standard method of approximation is Matching Pursuit. Matching Pursuit is a greedy algorithm originally designed for finite-dimensional signals, i.e., $\mathcal{H} = \mathbb{R}^d$. Following the notation of Problem (3) let $\bar{z}$ be the target vector we wish to approximate. In Matching Pursuit the first step is to pick an "atom" in $\{z_1, \ldots, z_n\}$ which captures most of $\bar{z}$ as measured by the magnitude of the inner product. After this first step the subsequent atoms are iteratively chosen according to which one captures more of the portion of $\bar{z}$ that hasn't been accounted for [17]. Note that just the first step of this algorithm requires to compute, for each z_i, the quantity $\langle \bar{z}, z_i \rangle = \frac{1}{n}\sum_{j \in [n]} \langle z_i, z_j \rangle$. Since we have n z_i's, the first step already takes $\Omega(n^2)$ inner product (kernel) evaluations, which is undesirable. A variant of matching pursuit specifically designed to approximate probability distributions through kernel means is kernel herding [18]. While [18] chooses the nonzero values of α_i to equal $1/k$, [19] proposes to use a line search to obtain nonuniform α_i values. In herding the complete kernel matrix is also computed, taking quadratic computational time. Another general common approach to sparse approximation, Basis Pursuit, has similar time complexity.

Several algorithms which focus specifically on the sparse KDE problem have been developed. In [20] a clustering method is used to approximate the KDE at a point by rejecting points which fail to belong to close clusters. In [21] a relevant subset of the data is chosen to minimize the L^2 error but at an expensive $O(n^2)$ cost. In [22], [23] a regression based approach is taken to estimate the KDE through its cumulative density function. These algorithms rely on the assumption that the kernel mean in question is a KDE, so cannot be generalized to other kernel means.

When the kernel mean is thought of as a mixture model, the model can be collapsed into a simpler one by reducing the number of its components through a similarity based merging procedure [24], [25], [26]. Since these methods necessitate the computation of all pairwise similarities, they present quadratic computational complexity. EM algorithms for this task result in similar computational requirements [27], [28].

A line of work that tries to speed up general kernel sums comes historically from n-body problems in physics, and makes use of fast multipole methods [29], [30]. The general idea behind these methods is to represent the kernel in question by a truncated series expansion, and then use a space partitioning scheme to group points, yielding an efficient way to approximate group-group or group-point interactions, effectively reducing the number of kernel evaluations. These methods are usually kernel-dependent. For the case of the Gaussian kernel, see [31], [32] for two different space partitioning methods. Note that space partitioning schemes may suffer considerably in high dimensional settings. Also, since the kernel function is truncated through its series expansion, the resulting approximation may not integrate to 1. Contrary to these methods, our approach can still yield a valid density (discussed below).

The efforts of rapidly approximating general kernel based quantities have led to the use of ϵ-samples, or coresets. To define ϵ-samples, first denote the data $A := \{x_1, \ldots, x_n\}$ and the kernel quantity of interest $Q(A, x)$, where x is some query point (for example, the KDE is $Q(A,x) = \frac{1}{n}\sum_{i\in[n]}\phi(x_i,x)$). An ϵ-sample is a set $A' \subset A$ such that, for every query point x, $Q(A,x)$ and $Q(A',x)$ differ by less than ϵ with respect to some norm. See [33], [34] for the KDE case with ℓ_∞ norm. For the KME using the RKHS norm, see [35]. Both cases allow for constructions of ϵ-samples in near linear time with respect to the data size and $1/\epsilon$. Notice that our approach has the advantage that it handles both the KDE and KME cases simultaneously, and that if desired it can yield a valid density as the approximation.

Although most of the literature seems to concentrate on the KDE, there have also been efforts to speed up computation time in problems involving the KME. As in the ϵ-sample approach above, many of these problems require the distance between KMEs in the RKHS, so they focus on speeding up this calculation. In [36], for example, a fast method is devised for the specific case of the maximum mean discrepancy statistic used for the two-sample test.

Computing the kernel mean at each of the original points $\{x_1,\dots,x_n\}$ can be thought of as a matrix vector multiplication, where the matrix in question is the kernel matrix. Therefore, an algebraic approach to this problem consists of choosing a suitable subset of the matrix columns and then approximating the complete matrix only through these columns. Among the most common of these is the Nyström method. In the Nyström method the kernel matrix K is approximated by the matrix $QW_r^+Q^T$, where Q is composed of a subset of the columns of K, indexed by $\mathcal{I}$, W is the matrix with entries K_{ij} for $(i,j)\in\mathcal{I}\times\mathcal{I}$, and W_r^+ is the best r-rank approximation to its pseudoinverse (see [37] for details). The columns composing Q are typically chosen randomly under some sampling distribution. See [38] for some examples of sampling distributions. As explained in Section IV-A, our approach is connected to the Nyström method and can be viewed as a particular scheme for column selection tailored to kernel means. The Nyström approximation of the kernel matrix is not the only one used though, and other algebraic approaches exist. In [39] for example, an interpolative decomposition of the kernel matrix is proposed.

In [40] a "coherence" based sparsification criterion is used in the context of one-class classification. The main idea is that each set of possible atoms $\{z_i | \alpha_i \neq 0\}$ can be quantified by the largest absolute value of the inner product between two different atoms. The method proposed requires the computation of the complete kernel matrix, and is therefore not suitable for our setting, which involves large data. The motivation for their coherence criterion, however, lies in the minimization of a bound on the approximation error. As seen in Section IV-B, we propose a similar bound as a starting point for our algorithm.

Contributions

We list a summary of contributions in this paper.

- We present a bound on the sparse approximation error based on a novel measure of incoherence.
- We recognize that for radial kernels, and more generally for any kernel such that $\|\phi(\cdot,x)\|$ is constant (which includes all translation invariant kernels), minimizing the bound is equivalent to solving an instance of the k-center problem. The solution to the k-center problem, in turn, can be approximated by a linear running time algorithm.
- Our method for approximating the KDE can be implemented so that the sparse kernel mean is a valid density function, which is important for some applications. Note that some alternative methods cannot be adapted to do so.
- Our method provides amortization of computational complexity since the calculation of the set $\mathcal{I}$ (introduced below) is only done once. Many subsequent calculations (e.g., kernel bandwidth search) can then be performed at a relatively small or negligible cost.
- Our method is versatile in that it addresses different types of kernel means under a common framework. In particular, it can be used to approximate both KMEs and KDEs at the same time.
- Our method provides a scheme to automatically select the sparsity level.
- We demonstrate the improved performance of the proposed method in three different applications, Euclidean embedding of probabilities (using both the KDE and the KME), class proportion estimation (using the KME), and clustering with the mean-shift algorithm (using the KDE), as well as on several additional benchmark datasets.

IV. Subset Selection and Incoherence-Based Bound

Let us now reformulate problem (3). Our approach will be to separate the problem into two parts: that of finding the set of indices i such that α_i is not zero, and that of finding the value of the nonzero α_i's. Letting $\mathcal{I}\subset[n]$ denote an index set, we can pose problem (3) as

$$\min_{\substack{\mathcal{I}\subseteq[n]\\|\mathcal{I}|=k}} \min_{(\alpha_i)_{i\in\mathcal{I}}} \|\bar{z} - \sum_{i\in\mathcal{I}}\alpha_i z_i\|^2 . \tag{4}$$

Note that the inner optimization problem is unconstrained and quadratic, and its solution, which for fixed $\mathcal{I}$ and k we denote by $\alpha_\mathcal{I}\in\mathbb{R}^k$, is

$$\alpha_\mathcal{I} = K_\mathcal{I}^{-1}\kappa_\mathcal{I},$$

where $K_\mathcal{I} = (\langle z_i, z_j\rangle)_{i,j\in\mathcal{I}}$ and $\kappa_\mathcal{I}$ is the k-dimensional vector with entries $\frac{1}{n}\sum_{j\in[n]}\langle z_j, z_l\rangle,\ l\in\mathcal{I}$.

Let $\alpha_\mathcal{I} = (\alpha_{\mathcal{I},i})_{i\in\mathcal{I}}$ and $z_\mathcal{I} = \sum_{i\in\mathcal{I}}\alpha_{\mathcal{I},i}z_i$. Then we can rewrite problem (3) as

$$\min_{\substack{\mathcal{I}\subseteq[n]\\|\mathcal{I}|=k}} \|\bar{z} - z_\mathcal{I}\| . \tag{5}$$

A. Connection to the Nyström Method

Before continuing to the approximate solution of problem (5), we briefly highlight its relationship to the Nyström method. Given a set $\mathcal{I}\subset[n]$, let K be the kernel matrix of $\{z_i \mid i\in[n]\}$, $K := (\langle z_i,z_j\rangle)_{i,j\in[n]}$, and $K_\mathcal{I}$ the kernel matrix of $\{z_i \mid i\in\mathcal{I}\}$, $K_\mathcal{I} := (\langle z_i,z_j\rangle)_{i,j\in\mathcal{I}}$. Also, let $Q_\mathcal{I}$ be the binary matrix such that $KQ_\mathcal{I}$ is composed of the columns of K corresponding to $\mathcal{I}$. Then we can rewrite $\alpha_\mathcal{I}$ and $K_\mathcal{I}$ as $\alpha_\mathcal{I} = (Q_\mathcal{I}^TKQ_\mathcal{I})^{-1}Q_\mathcal{I}^TK\mathbf{1}_n$ and $K_\mathcal{I} = Q_\mathcal{I}^TKQ_\mathcal{I}$, where $\mathbf{1}_n$ denotes the vector in $\mathbb{R}^n$ with entries $1/n$. By doing so, we can express the objective of (5) as

$$\begin{aligned}\|\bar{z} - z_{\mathcal{I}}\|^2 &= \mathbf{1}_n^T \left(K - KQ_{\mathcal{I}}K_{\mathcal{I}}^{-1}Q_{\mathcal{I}}^T K^T\right)\mathbf{1}_n \\ &= \mathbf{1}_n^T \left(K - \tilde{K}_{\mathcal{I}}\right)\mathbf{1}_n.\end{aligned}$$

where $\tilde{K}_{\mathcal{I}} := KQ_{\mathcal{I}}K_{\mathcal{I}}^{-1}Q_{\mathcal{I}}^T K^T$. We recognize $\tilde{K}_{\mathcal{I}}$ as the Nyström matrix from the Nyström method [38], which is the only term dependent on $\mathcal{I}$ in the objective. Therefore, our work can be interpreted from the Nyström perspective: choose suitable columns of K and approximate K through the Nyström matrix. The main difference is that the resulting approximation is assessed using the induced norm of the inner product space where the z_i's reside, instead of the commonly used spectral and Frobenius norms.

B. An Incoherence-Based Sparse Approximation Bound

We now present our proposed algorithm to approximate the solution of problem (5). Our strategy is to find an upper bound on $\|\bar{z} - z_{\mathcal{I}}\|$ which is dependent on $\mathcal{I}$ and then find the $\mathcal{I}$ that minimizes the bound. First, we present a lemma which will aid us in finding the bound.

Lemma 1: Let $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ be an inner product space. Let S be a finite dimensional subspace of $\mathcal{H}$ and P_S the projection onto S. For any $z_0 \in \mathcal{H}$

$$\|P_S z_0\| = \max_{z \in S, \|z\|=1} \langle z_0, z\rangle.$$

Proof: First note that since S is finite dimensional, by the Projection Theorem $z_0 - P_S z_0$ is orthogonal to S. Now, for any $z \in S$ with $\|z\| = 1$, we have

$$\begin{aligned}\langle z_0, z\rangle &= \langle P_S z_0 + (z_0 - P_S z_0), z\rangle \\ &= \langle P_S z_0, z\rangle + \langle z_0 - P_S z_0, z\rangle \\ &= \langle P_S z_0, z\rangle \\ &\le \|P_S z_0\|\,\|z\| = \|P_S z_0\|,\end{aligned}$$

where we have used the Cauchy-Schwartz inequality. To confirm the existence of a vector z which makes it an equality and therefore reaches the maximum, just let $z = P_S z_0/\|P_S z_0\|$. ■

We can now present the theorem which will be the basis for our minimization approach. First, define

$$\nu_{\mathcal{I}} := \min_{j\notin\mathcal{I}} \max_{i\in\mathcal{I}} \langle z_i, z_j\rangle,$$

which we can think of as a measure of the "incoherence" of $\{z_i \mid i \in \mathcal{I}\}$. It is now possible to establish a bound:

Theorem 1: Assume that for some $C > 0$ $\langle z_i, z_i\rangle = C$ $\forall i \in [n]$. Then for every $\mathcal{I} \subseteq [n]$,

$$\|\bar{z} - z_{\mathcal{I}}\| \le \left(1 - \frac{|\mathcal{I}|}{n}\right)\sqrt{\frac{1}{C}\left(C^2 - \nu_{\mathcal{I}}^2\right)}.$$

Proof: The beginning of this proof is similar to the one in [40]. Let $S_{\mathcal{I}} := \operatorname{span}(\{z_i \mid i \in \mathcal{I}\})$ and denote $P_{S_{\mathcal{I}}}$ the projection operator onto $S_{\mathcal{I}}$ and I the identity operator. We have

$$\begin{aligned}\|\bar{z} - z_{\mathcal{I}}\| &= \|\bar{z} - P_{S_{\mathcal{I}}}\bar{z}\| = \frac{1}{n}\Big\|\sum_{i\in[n]}(I - P_{S_{\mathcal{I}}})z_i\Big\| \\ &\le \frac{1}{n}\sum_{i\in[n]}\|(I - P_{S_{\mathcal{I}}})z_i\| = \frac{1}{n}\sum_{i\notin\mathcal{I}}\|(I - P_{S_{\mathcal{I}}})z_i\|\end{aligned}$$

where we have used the triangle inequality, and the last equality is due to the fact that $z_i = P_{S_{\mathcal{I}}}z_i$ when $z_i \in S_{\mathcal{I}}$.

Now, since $(z_i - P_{S_{\mathcal{I}}}z_i) \perp P_{S_{\mathcal{I}}}z_i$, we can use Pythagoras' Theorem in $\mathcal{H}$ to get $\|z_i - P_{S_{\mathcal{I}}}z_i\|^2 = \|z_i\|^2 - \|P_{S_{\mathcal{I}}}z_i\|^2$.

By Lemma 1, $\|P_{S_{\mathcal{I}}}z_i\| = \max_{z\in S_{\mathcal{I}},\,\|z\|=1}\langle z_i, z\rangle$. Therefore, for $i \notin \mathcal{I}$,

$$\begin{aligned}\|P_{S_{\mathcal{I}}}z_i\| &= \frac{1}{\sqrt{C}}\max_{z\in S_{\mathcal{I}},\|z\|=\sqrt{C}}\langle z_i, z\rangle \\ &\ge \frac{1}{\sqrt{C}}\max_{\ell\in\mathcal{I}}\langle z_i, z_\ell\rangle \\ &\ge \frac{1}{\sqrt{C}}\min_{j\notin\mathcal{I}}\max_{\ell\in\mathcal{I}}\langle z_j, z_\ell\rangle = \frac{1}{\sqrt{C}}\nu_{\mathcal{I}}.\end{aligned}$$

Thus, for $i \notin \mathcal{I}$,

$$\|z_i\|^2 - \|P_{S_{\mathcal{I}}}z_i\|^2 \le C - \frac{\nu_{\mathcal{I}}^2}{C}$$

and finally

$$\|\bar{z} - z_{\mathcal{I}}\| \le \frac{1}{n}\sum_{i\notin\mathcal{I}}\sqrt{C - \frac{\nu_{\mathcal{I}}^2}{C}} = \left(1 - \frac{|\mathcal{I}|}{n}\right)\sqrt{\frac{1}{C}(C^2 - \nu_{\mathcal{I}}^2)}.$$

■

C. Application to Kernel Means

We now apply the previous result in the context of approximating a kernel mean based on a radial kernel. Recall that, in the kernel mean setting, $z_i = \phi(\cdot, x_i)$ and $\langle\phi(\cdot, x_i), \phi(\cdot, x_j)\rangle = g(\|x_i - x_j\|_2)$, where ϕ is a radial kernel, $\{x_1, \ldots, x_n\} \subset \mathbb{R}^d$, and g is strictly decreasing as in Definition 3. Also note that for any radial kernel the assumption in Theorem 1 is satisfied, since $\langle\phi(\cdot, x_i), \phi(\cdot, x_i)\rangle = g(0) = C > 0$. Here $\bar{f}$ and $f_{\mathcal{I}}$ are defined in an analogous way to $\bar{z}$ and $z_{\mathcal{I}}$, with $\bar{f}$ being a kernel mean and $f_{\mathcal{I}}$ its sparse approximation. The following corollary follows directly from Theorem 1.

Corollary 1: Let ϕ be a radial kernel, with $\langle\phi(\cdot, x), \phi(\cdot, x)\rangle_{\mathcal{H}} = C$ $\forall x \in \mathbb{R}^d$. Then for every $\mathcal{I} \subseteq [n]$,

$$\|\bar{f} - f_{\mathcal{I}}\| \le \left(1 - \frac{|\mathcal{I}|}{n}\right)\sqrt{\frac{1}{C}\left(C^2 - \nu_{\mathcal{I}}^2\right)}.$$

■

For the case of symmetric positive definite kernels, with $\mathcal{H}$ the corresponding RKHS, bounding the $\mathcal{H}$ norm implies bounding the L^∞ norm, as stated in the following corollary.

Corollary 2: Let ϕ be a symmetric positive definite kernel with associated RKHS $\mathcal{H}$, with $\langle\phi(\cdot, x), \phi(\cdot, x)\rangle_{\mathcal{H}} = C$ $\forall x \in \mathbb{R}^d$. Then for every $\mathcal{I} \subseteq [n]$,

$$\|\bar{f} - f_{\mathcal{I}}\|_\infty \le \left(1 - \frac{|\mathcal{I}|}{n}\right)\sqrt{C^2 - \nu_{\mathcal{I}}^2}.$$

Proof:

$$\begin{aligned}\left\|\bar{f}-f_{\mathcal{I}}\right\|_{\infty} &= \max_{x}\left|\left(\bar{f}-f_{\mathcal{I}}\right)(x)\right| \\ &= \max_{x}\left|\langle\bar{f}-f_{\mathcal{I}},\phi(\cdot,x)\rangle\right| \\ &\leq \max_{x}\left\|\bar{f}-f_{\mathcal{I}}\right\|_{\mathcal{H}}\|\phi(\cdot,x)\|_{\mathcal{H}} \\ &= \sqrt{C}\left\|\bar{f}-f_{\mathcal{I}}\right\|_{\mathcal{H}},\end{aligned}$$

where the second line is due to the reproducing property and the third to the Cauchy-Schwarz inequality. ■

For some applications, such as density estimation, one may desire the approximation to belong to $\Delta := \{\sum_{i\in\mathcal{I}}\alpha_i z_i \mid \sum\alpha_i = 1, \alpha_i \geq 0\}$. A similar bound can be derived, but is not as tight as the previous ones. It also suggests, however, the maximization of the term $\nu_{\mathcal{I}}$. In particular, we have that, under the assumptions of Corollary 1, and letting f_Δ be the projection of $\bar{f}$ onto Δ:

$$\left\|\bar{f}-f_\Delta\right\| \leq \sqrt{2\left(C-\left(1-\frac{|\mathcal{I}|}{n}\right)\nu_{\mathcal{I}}\right)}.$$

Details are shown in the Appendix.

V. Bound Minimization Via k-center Algorithm

The bound in the previous corollaries can be minimized by maximizing the term $\nu_{\mathcal{I}}$. We now present a procedure to accomplish this for the case of radial kernels. Let ϕ be a radial kernel and define the set $\mathcal{I}^*$ as

$$\mathcal{I}^* := \arg\min_{\substack{\mathcal{I}\subseteq[n]\\|\mathcal{I}|=k}}\max_{j\notin\mathcal{I}}\min_{i\in\mathcal{I}}\|x_i-x_j\|.$$

Then, since $\langle\phi(\cdot,x_i),\phi(\cdot,x_j)\rangle = g(\|x_i-x_j\|_2)$ for g strictly decreasing, $\mathcal{I}^*$ also maximizes $\nu_{\mathcal{I}} = \min_{j\notin\mathcal{I}}\max_{i\in\mathcal{I}} g(\|x_i - x_j\|)$. Therefore, $\mathcal{I}^*$ is the set that minimizes the bound in Theorem 1. We have translated a problem involving inner products of functions to a problem involving distances between points in $\mathbb{R}^d$.

The problem of finding $\mathcal{I}^*$ is known as the k-center problem. To pose the k-center problem more precisely, we make a few definitions. For a fixed $\mathcal{I}$, let $X_{\mathcal{I}} = \{x_i \mid i\in\mathcal{I}\}$ and $Y_{\mathcal{I}} = \{x_j \mid j\notin\mathcal{I}\}$, and for all $x_j\in Y_{\mathcal{I}}$ define its distance to $X_{\mathcal{I}}$ as $d(x_j,X_{\mathcal{I}}) = \min_{x_i\in X_{\mathcal{I}}}\|x_i-x_j\|$. Furthermore, let $W(X_{\mathcal{I}}) = \max_{x_j\in Y_{\mathcal{I}}} d(x_j,X_{\mathcal{I}})$. Then, the k-center problem is that of finding the set $\mathcal{I}$ of size k for which $W(X_{\mathcal{I}})$ is minimized.

The k-center problem is known to be NP-complete [41]. However, there exists a greedy 2-approximation algorithm [42] which produces a set $\mathcal{I}_k$ such that $W(X_{\mathcal{I}_k}) \leq 2W(X_{\mathcal{I}^*})$. This algorithm is optimal in the sense that under the assumption that P≠NP there is no ρ-approximation algorithm with $\rho < 2$ [43]. The algorithm is described in Fig. 1, and as can be seen, it has a linear time complexity in the size of the data n. In particular, the algorithm runs in $O(nkd)$ time.

To relate the output of the algorithm back to the bound of the theorem, note that $\nu_{\mathcal{I}} = g(W(X_{\mathcal{I}}))$. Since the k-center algorithm guarantees that $W(X_{\mathcal{I}_k}) \leq 2W(X_{\mathcal{I}^*})$, in the most general case we can say that $\sqrt{C^2-\nu_{\mathcal{I}_k}^2} = \sqrt{C^2-g(W(X_{\mathcal{I}_k}))^2} \leq \sqrt{C^2-g(2W(X_{\mathcal{I}^*}))^2}$. Knowing more about the form of g yields more information. For example, for the Gaussian kernel we have

$$\begin{aligned}&\sqrt{\frac{1}{C}\left(C^2-\nu_{\mathcal{I}_k}^2\right)} \leq \sqrt{\frac{1}{C}\left(C^2-\frac{1}{C^6}\nu_{\mathcal{I}^*}^8\right)} \\ &\leq \sqrt{\frac{1}{CC^6}\left(C^4-\nu_{\mathcal{I}^*}^4\right)\left(C^4+\nu_{\mathcal{I}^*}^4\right)} \\ &\leq \sqrt{\frac{2C^4}{CC^6}\left(C^2-\nu_{\mathcal{I}^*}^2\right)\left(C^2+\nu_{\mathcal{I}^*}^2\right)} \leq 2\sqrt{\frac{1}{C}\left(C^2-\nu_{\mathcal{I}^*}^2\right)}.\end{aligned}$$

```
input $x_1,\dots,x_n,k$
$X \longleftarrow \varnothing$
$Y \longleftarrow \{x_1,\dots,x_n\}$
Choose randomly a first index $u\in[n]$
$X \longleftarrow X\cup\{x_u\}$
$Y \longleftarrow Y\backslash\{x_u\}$
while $|X| < k$ do
  Choose the element $y\in Y$ for which $d(y,X)$ is maximized
  $X \longleftarrow X\cup\{y\}$
  $Y \longleftarrow Y\backslash\{y\}$
end while
output $\mathcal{I}_k = \{i\in[n] \mid x_i\in X\}$
```

Fig. 1. A linear time 2-approximation algorithm for the k-center problem.

A. Generalization to Nonradial Kernels

The preceding argument extends to certain nonradial kernels. For example, consider kernels satisfying $\langle\phi(\cdot,x),\phi(\cdot,x')\rangle = g(M(x,x'))$, where M is a non-Euclidean metric on $\mathbb{R}^d$. Examples include the Gaussian kernel with anisotropic covariance where M is a Mahalanobis distance, or a type of Laplacian kernel where M is the ℓ_1 distance. The k-center algorithm described previously applies in these settings as well, where distance in the algorithm is computed using M.

Another example of a nonradial kernel is a *discriminative kernel* for classification problems. Let x denote a feature vector and $y\in\{-1,1\}$ its label. If ϕ is a kernel on $\mathbb{R}^d$, then $\psi((x,y),(x',y')) = yy'\phi(x,x')$ is a kernel on $\mathbb{R}^d\times\{-1,1\}$. A kernel mean based on this kernel is a well-known classification algorithm [44]. For the discriminative kernel, the problem of maximizing the incoherence reduces to a variation of the k-center problem: find k_1 points in one class and k_{-1} points in the other class such that $k_1+k_{-1}=k$ and the maximum distance of a point to its nearest representative *in the same class* is minimized. The k-center algorithm described previously can be easily adapted to solve this problem.

More generally, let ϕ be any kernel such that $\|\phi(\cdot,x)\|^2 = C$ is independent of x. This includes any translation invariant kernel. Then

$$\langle\phi(\cdot,x),\phi(\cdot,x')\rangle = \frac{1}{2}(2C-\|\phi(\cdot,x)-\phi(\cdot,x')\|^2).$$

Hence, the problem of solving

$$\max_{\substack{\mathcal{I}\subseteq[n]\\|\mathcal{I}|=k}} \min_{j\notin\mathcal{I}} \max_{i\in\mathcal{I}} \langle \phi(\cdot, x_i), \phi(\cdot, x_j)\rangle$$

reduces to solving

$$\min_{\substack{\mathcal{I}\subseteq[n]\\|\mathcal{I}|=k}} \max_{j\notin\mathcal{I}} \min_{i\in\mathcal{I}} \|\phi(\cdot, x_i) - \phi(\cdot, x_j)\| .$$

In other words, it suffices to solve the k-center problem in the kernel feature space $\mathcal{H}$, using the induced $\mathcal{H}$ norm, which can be calculated efficiently using the kernel. Once again, the same k-center algorithm can be applied, where distances are now computed as $\|\phi(\cdot, x_i) - \phi(\cdot, x_j)\| = \sqrt{2C - 2\phi(x_i, x_j)}$. We also note that Corollaries 1 and 2 also hold in this generalized setting. It is worth noting that for the case of radial kernels the k-center algorithm can be used in both the feature space as in Euclidean space, which will potentially yield two different results. We have not empirically validated the nonradial kernel case.

B. Computation of $\alpha_{\mathcal{I}}$ and Auto-Selection of k

The k-center algorithm allows us to find the set $\mathcal{I}$ on which our approximation will be based. After finding $\mathcal{I}$ we can determine the optimal coefficients $\alpha_{\mathcal{I}}$. Since the main computational burden is in the selection of $\mathcal{I}$, we now have the freedom to explore different values of $\alpha_{\mathcal{I}}$ in a relatively small amount of time. For example, we can compute $\alpha_{\mathcal{I}}$ for each of several possible kernel bandwidths σ.

The optimal way to compute $\alpha_{\mathcal{I}}$ depends on the application. If the user has a good idea of what the value of k is, then a fast way to compute $\alpha_{\mathcal{I}}$ for that specific value is to apply their preferred method to solve the equation $K_{\mathcal{I}}\alpha_{\mathcal{I}} = \kappa_{\mathcal{I}}$. For example, since for symmetric positive definite kernels the kernel matrix is positive semi-definite, the preconditioned conjugate gradient method can be used to quickly obtain $\alpha_{\mathcal{I}}$ to high accuracy. This approach has the advantages of being simple and fast.

A further advantage of our method is evident when the user can accept a maximum tolerance value of k, say k_{max}, but would prefer to stop at a value $k_0 \leq k_{max}$ that performs about as well as k_{max}. To do this, at iteration $m \geq 1$ in the k-center algorithm we compute $\alpha_{\mathcal{I}_m}$ right after computing $\mathcal{I}_m$, which provides a record of all the $\alpha_{\mathcal{I}_j}$ for $1 \leq j \leq k_0$. To find k_0, we use the information from the computed coefficients to form an error indicator and stop when some error threshold is crossed. Before showing what these error indicators are, we first provide an update rule to efficiently compute the α coefficients at each iteration step.

Let $\mathcal{I}_m$ be the set of the first m elements chosen by the k-center algorithm, and let $\alpha_{\mathcal{I}_m}$, $K_{\mathcal{I}_m}$ and $\kappa_{\mathcal{I}_m}$ be obtained by using $\mathcal{I}_m$. If we increase the number of components to $m+1$, then as shown in [40] we have

$$K_{\mathcal{I}_{m+1}} = \begin{bmatrix} K_{\mathcal{I}_m} & b \\ b^T & \phi(x_{j_{m+1}}, x_{j_{m+1}}) \end{bmatrix}$$

where x_{j_ℓ} is the ℓ^{th} element selected by the k-center algorithm, and $b = (\phi(x_{j_{m+1}}, x_i))_{i\in\mathcal{I}_m}$. The resulting update rule for the inverse is

$$K_{\mathcal{I}_{m+1}}^{-1} = \begin{bmatrix} K_{\mathcal{I}_m}^{-1} & 0 \\ 0 & 0 \end{bmatrix} + q_0(qq^T)$$

where $q_0 = 1/(\phi(x_{j_{m+1}}, x_{j_{m+1}}) - b^T K_{\mathcal{I}_m}^{-1} b)$ and $q = [-b^T K_{\mathcal{I}_m}^{-1} \quad 1]^T$. From here the user can now compute $\alpha_{\mathcal{I}_{m+1}}$ by multiplying $K_{\mathcal{I}_{m+1}}^{-1}$ with

$$\kappa_{\mathcal{I}_{m+1}} = \begin{bmatrix} \kappa_{\mathcal{I}_m} \\ \frac{1}{n}\sum_{i=1}^n \phi(x_{j_{m+1}}, x_i) \end{bmatrix}.$$

Assuming we stop at k_{max}, the time complexity for computing all the $\alpha_{\mathcal{I}_m}$'s is $O(k_{max}^3)$ and the necessary memory $O(k_{max}^2)$.

Note that our incremental approach to construct $\alpha_{\mathcal{I}}$ does assume that the kernel matrix $K_{\mathcal{I}}$ is full rank, since it does compute $K_{\mathcal{I}}^{-1}$ explicitly, and not the pseudoinverse (as opposed to, say, the Nyström method). For Gaussian and similar kernels, $K_{\mathcal{I}}$ is positive definite assuming the x_i's are distinct. Rank deficiency results from selecting centers that are very close to each other, however, the k-center algorithm does the opposite and selects elements far apart from each other, which supports the assumption of a full rank matrix $K_{\mathcal{I}}$.

To automatically stop at some $k_0 \leq k_{max}$ we need a stopping criterion based on some form of error. We propose the following: using the notation of problem (5) we have that

$$\begin{aligned}\|\bar{z}-z_{\mathcal{I}}\|^2 &= \left\langle \frac{1}{n}\sum_{\ell\in[n]} z_\ell, \frac{1}{n}\sum_{\ell'\in[n]} z_{\ell'} \right\rangle \\ &\quad - 2\left\langle \frac{1}{n}\sum_{\ell\in[n]} z_\ell, \sum_{i\in\mathcal{I}} \alpha_{\mathcal{I},i} z_i \right\rangle + \left\langle \sum_{i\in\mathcal{I}} \alpha_{\mathcal{I},i} z_i, \sum_{j\in\mathcal{I}} \alpha_{\mathcal{I},j} z_j \right\rangle \\ &= \|\bar{z}\|^2 - 2\cdot \sum_{i\in\mathcal{I}} \alpha_{\mathcal{I},i} \cdot \frac{1}{n}\sum_{\ell\in[n]} \langle z_\ell, z_i\rangle + \alpha_{\mathcal{I}}^T K_{\mathcal{I}} \alpha_{\mathcal{I}} \\ &= \|\bar{z}\|^2 - \alpha_{\mathcal{I}}^T \kappa_{\mathcal{I}}.\end{aligned}$$

Since $\|\bar{z}\|^2$ is a constant independent of $\mathcal{I}$, we can avoid its $O(n^2)$ computation and only use the quantities $E_{|\mathcal{I}|} := -\alpha_{\mathcal{I}}^T \kappa_{\mathcal{I}}$ as error indicators. Note that E_t is nonincreasing with respect to t. Based on this we choose k_0 to be the first value at which some relative error is small. In this paper we use the test

$$\frac{|E_{k_0-1} - E_{k_0}|}{|E_1 - E_{k_0}|} \leq \epsilon$$

for some small ϵ. The overall complexity is then reduced to $O(nk_0 d + k_0^3 d)$.

A further consideration for computing $\alpha_{\mathcal{I}}$ should be made if the result is desired to be a probability density function. In this case a k-dimensional $\alpha_{\mathcal{I}}$ can be projected into the simplex $\Delta^{k-1} := \{\nu \in \mathbb{R}^k \,|\, \sum_{i=1}^k \nu_i = 1, \nu_i \geq 0 \ \forall\ 1 \leq i \leq k\}$ after being obtained by any of the discussed methods (see [45]). This extra step takes negligible time with respect to the rest of the computations. Alternatively, a quadratic program which takes into account the constraints of non-negativity and $\sum_{i=1}^k \alpha_{\mathcal{I},i} = 1$ can be solved.

A Matlab implementation of the complete Sparse Kernel Mean procedure can be found at [2].

VI. Experiments: Speeding Up Existing Kernel Mean Methods

We have implemented our approach in three specific machine learning tasks that require the computation and evaluation of a mean of kernels. In the first of these, we apply our algorithm to the task of dimensionality reduction. In the second, we use it in the setting of class proportion estimation. In the third, we explore its performance when used as part of the mean shift algorithm. Finally, for 11 benchmark datasets we compare the performance of our approach to four other similar methods.

In the following we refer to our algorithm or to the resulting kernel mean as SKM (for Sparse Kernel Mean). We now provide a detailed description of each task and relevant results. The implementation has been done in Matlab.

A. Euclidean Embedding of Distributions

In this experiment we embed probability distributions in a lower dimensional space for the purpose of visualization. Given a collection of N distributions $\{P_1, \dots, P_N\}$, the procedure consists of creating a dissimilarity matrix for some notion of dissimilarity among these distributions and then performing a dimensionality reduction method. We consider two cases. In the first case the dissimilarity matrix will be the distance between the kernel mean embeddings of the distributions in the RKHS (KME case), while in the second case it will be the (symmetrized) KL divergence between KDEs (KDE case). For dimensionality reduction we will use ISOMAP [46]. In the setup we have access to each of N distributions $\{P_1, \dots, P_N\}$ through samples drawn from those distributions. The sample drawn from the ℓ^{th} distribution is denoted $\{x_i^{(\ell)}\}_{i=1}^{n_\ell}$.

Notice that in the KDE case, in order to compute the KL divergence it is necessary to obtain a valid density function. By choosing the coefficients as described in Section V-B, the resulting sparse approximation is a density function.

Let us start with the KME case, in which the dissimilarity is the difference between the distributions' KMEs. The first task is to estimate the KME using some symmetric positive definite kernel ϕ. For the ℓ^{th} distribution, the empirical estimate of its KME is

$$\widehat{\Psi}(P_\ell) = \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} \phi(\cdot, x_i^{(\ell)}),$$

with a sparse approximation

$$\widehat{\Psi}_0(P_\ell) = \sum_{i \in \mathcal{I}^{(\ell)}} \alpha_i^{(\ell)} \phi(\cdot, x_i^{(\ell)}),$$

for some set $\mathcal{I}^{(\ell)}$ and $\{\alpha_i^{(\ell)} | \alpha_i^{(\ell)} \in \mathbb{R}\}$, where the α coefficients have been computed according to the update method described in Section V-B.

Given all the KMEs, we can now construct a distance matrix. Let $\mathcal{H}$ be the RKHS of ϕ. We can use the distance induced by

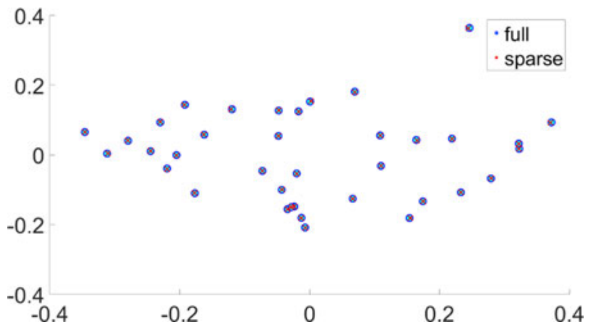

Fig. 2. 2-dimensional representation of flow cytometry data - KME case. Each point represents a patient's distribution. The embeddings were obtained by applying ISOMAP to distances in the RKHS.

the RKHS to create the matrix D, with entries

$$\begin{aligned} D_{\ell,\ell'} &:= \left\| \widehat{\Psi}(P_\ell) - \widehat{\Psi}(P_{\ell'}) \right\|_{\mathcal{H}} \\ &= \left[\frac{1}{n_\ell^2} \sum_{i,j} \phi(x_i^{(\ell)}, x_j^{(\ell)}) - 2 \frac{1}{n_\ell n_{\ell'}} \sum_{i,j} \phi(x_i^{(\ell)}, x_j^{(\ell')}) \right. \\ &\quad \left. + \frac{1}{n_{\ell'}^2} \sum_{i,j} \phi(x_i^{(\ell')}, x_j^{(\ell')}) \right]^{1/2}. \end{aligned}$$

We similarly define D_0 based on the sparse KMEs. With such matrices ISOMAP can now be performed to visualize the distributions in, say, $\mathbb{R}^2$.

Note that if the samples from P_ℓ and $P_{\ell'}$ have n_ℓ and $n_{\ell'}$ points, then $D_{\ell,\ell'}$ takes $\Theta(n_\ell^2 + n_\ell n_{\ell'} + n_{\ell'}^2)$ time to compute. Since we need all the pairwise distances, we need $\Theta(N^2)$ such computations. A sparse approximation of the KMEs of P_ℓ and $P_{\ell'}$ of sizes k_ℓ and $k_{\ell'}$ would instead yield a computation of $\Theta(k_\ell^2 + k_\ell k_{\ell'} + k_{\ell'}^2)$ for each entry. Assuming all samples have the same size n, and the sparse approximation size is k, then the computation of the distance matrix is reduced from $\Theta(N^2 n^2)$ to $\Theta(N^2 k^2)$.

Inspired by the work of [47], we have performed these experiments on flow cytometry data from $N = 37$ cancer patients, with sample sizes ranging from 8181 to 108343. We have used the Gaussian kernel, chosen $\mathcal{H}$ to be its RKHS, and computed the bandwidth based on the 'iqr' scale option in R's KernSmooth package. That is, we have computed the interquartile range of the data, averaged over each dimension, and divided by 1.35. After the embedding has been done, we have performed Procrustes analysis on the points so as to account for possible translation, scaling, and rotation.

To determine the maximum size k_ℓ of each sparse representation, we recall that the SKM procedure takes $O(n_\ell k_\ell + k_\ell^3)$ kernel evaluations, so in order to respect the $n_\ell k_\ell$ factor, we have chosen a small multiple of $\sqrt{n_\ell}$ for k_ℓ. In this case we picked k_ℓ to be the largest integer smaller than $3\sqrt{n_\ell}$ for each ℓ. We have implemented the auto-selection scheme described in Section V-B. The results for the case of $\epsilon = 10^{-12}$ are shown in Fig. 2 and Table I. As a comparison, we also compute an alternative D_0 based on kernel herding (see Section III) and plot

TABLE I
TIME COMPARISON FOR THE EUCLIDEAN EMBEDDING OF THE FLOW CYTOMETRY DATASET - KME CASE

	Approximation	D computation	Total
Full	0	3.04hrs	3.04hrs
SKM	1.29mins	1.6s	1.31mins
HERD	30.38mins	1.8s	30.42mins

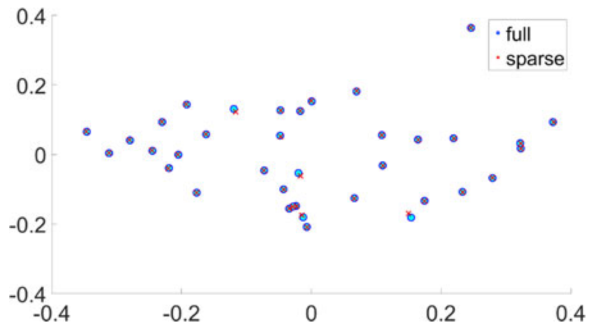

Fig. 3. 2-dimensional representation of flow cytometry data - KME case. D_0 was found through kernel herding.

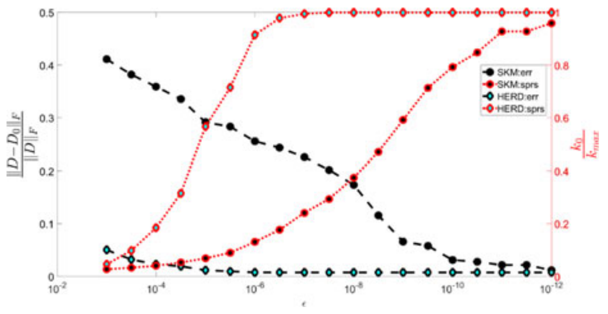

Fig. 4. The relative error and sparsity incurred by the SKM-based and HERD-based matrices D_0 as a function of ϵ, KME case. Not included is the computation time, SKM achieves an error of 10^{-12} in 1.3 minutes, while HERD takes more than 30 minutes.

the result in Fig. 3. Note that although for this fixed ϵ HERD seems to fit better the projection resulting from using the full kernel mean, as seen in Table I it takes 30 minutes to do so. For the SKM an almost identical projection can be generated in just over a minute.

Although k_ℓ is the largest allowed sparsity, each algorithm stops at some $k_{0\ell} \leq k_\ell$. The values $\frac{k_{0\ell}}{k_\ell}$ averaged over all ℓ's are shown in Fig. 4. To determine how well D_0 approximates D, we have plotted the relative error $\frac{\|D-D_0\|_F}{\|D\|_F}$ for different values of ϵ. The result is shown in Fig. 4.

The KDE case is similar. The dissimilarity matrix is composed of the symmetrized KL divergence between the KDEs of the distributions, defined as $d_{KL}(p,q) := D_{KL}(p\|q) + D_{KL}(q\|p)$, where D_{KL} indicates the KL divergence. For the ℓ^{th} distribution, its KDE is

$$\widehat{f}_\ell = \frac{1}{n_\ell}\sum_{i=1}^{n_\ell} \phi(\cdot, x_i^{(\ell)}),$$

with a sparse approximation

$$\widehat{f}_{0\ell} = \sum_{i\in\mathcal{I}^{(\ell)}} \alpha_i^{(\ell)} \phi(\cdot, x_i^{(\ell)}).$$

for some set $\mathcal{I}^{(\ell)}$ and $\{\alpha_i^{(\ell)} | \alpha_i^{(\ell)} \geq 0, \quad \sum_i \alpha_i^{(\ell)} = 1\}$, which has again been calculated according to the update method described in Section V-B. Note that the KL divergence requires two density functions as input. We achieve this by projecting onto the simplex as indicated in V-B. As in the KME case, we construct the dissimilarity matrix $(D_{\ell,\ell'}) := d_{KL}(\widehat{f_\ell}, \widehat{f_{\ell'}})$. Figures and Tables analogous to those for the KME case are shown in the appendix, with similar results.

Figure 2 shows us that the resulting embedded points using the sparse approximation keep the structure as of those using the full kernel mean. Notice also from Table I that the sparse approximation is many times faster than the full computation (about 400 times faster). Furthermore, the main computational investment is made in finding the elements of the sets $\mathcal{I}^{(\ell)}$, since the subsequent computation of D_0 is of negligible time.

In summary, for the case $\epsilon = 10^{-12}$, SKM takes only 1.3 minutes and the resulting embedding is almost identical to the full one, just as in Figure 3 shows for the herding case (which takes 30 minutes to compute). The SKM embedding is practically indistinguishable from that of the full one, being about 400 times faster than the full one and about 20 times faster than HERD.

B. Class Proportion Estimation

In this problem we are presented with labeled training data drawn from N distributions $\{P_1, \ldots, P_N\}$ and with further testing data drawn from a mixture of these distributions $P_0 = \sum_{i=1}^N \pi_i P_i$, where $\pi_i \geq 0$ and $\sum_i \pi_i = 1$. Each P_i represents a class in a multiclass classification problem and the goal is to estimate the mixture proportions $\{\pi_1, \ldots, \pi_N\}$ of each class in the unlabeled data set represented by P_0 (see [48], [49], [50]).

To do so we let $\widehat{\Psi}(P_\ell)$ represent the KME of P_ℓ for $0 \leq \ell \leq N$. We then find the proportions $\{\hat{\pi}_i\}_{i=1}^N$ that minimize the distance

$$\|\widehat{\Psi}(P_0) - \sum_{i=1}^N \pi_i \widehat{\Psi}(P_i)\|^2_{\mathcal{H}},$$

where $\mathcal{H}$ is the RKHS of the kernel used to construct the KME. By setting the derivative to zero the optimal vector of proportions $\hat{\pi}_- := [\hat{\pi}_1, \ldots, \hat{\pi}_{N-1}]^T$, subject to $\sum_{i=1}^N \hat{\pi}_i = 1$ but not to $\hat{\pi}_i \geq 0$, satisfies

$$\hat{D}\hat{\pi}_- = \hat{e},$$

where

$$\hat{D}_{ij} = \left\langle \widehat{\Psi}(P_i) - \widehat{\Psi}(P_N), \widehat{\Psi}(P_j) - \widehat{\Psi}(P_N)\right\rangle_{\mathcal{H}}$$

and

$$\hat{e}_i = \left\langle \widehat{\Psi}(P_i) - \widehat{\Psi}(P_N), \widehat{\Psi}(P_0) - \widehat{\Psi}(P_N)\right\rangle_{\mathcal{H}}.$$

From here we can define

$$\hat{\pi} := \begin{bmatrix} \hat{\pi}_- \\ 1 - \sum_{i=1}^{N-1} \hat{\pi}_i \end{bmatrix}.$$

A parallel approach, using the KDE instead of the KME is shown in [49]. In that case the distance in $\mathcal{H}$ was changed to the L^2 distance.

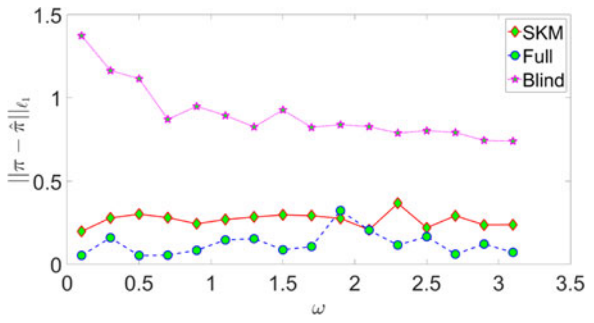

Fig. 5. Class Proportion Estimation. ℓ_1 error of estimated proportions over a range of concentration parameters.

Notice we have not enforced the constraint $\hat{\pi}_i \geq 0$, for $1 \leq i \leq N$. To do so a quadratic program can be set. For most of our simulations we did not encounter the necessity to do so, due to the fact that the true proportions we are approximating are also nonnegative, although this is an empirical observation without theoretical support. Therefore, for the few cases for which $\hat{\pi}$ lied outside of the simplex, we have projected onto it as described in [45].

We have used the handwritten digits data set MNIST, obtained from [51], which contains 60,000 training images and 10,000 testing images, approximately evenly distributed among its 10 classes (see [52] for details). We have only used the first five digits.

We present a comparison of the performance, measured by the ℓ_1 distance between the true π and the estimate $\hat{\pi}$, of the sparse KME compared to the full KME. We have done this for different values of π, meaning different locations of π inside the simplex. To do so, we sampled π from the simplex using the Dirichlet distribution with different concentration parameter ω. As a reminder to the reader, a small value of ω implies sparse values of π are most probable, $\omega = 1$ means any value of π is equally probable, and $\omega > 1$ means values of π for which all its entries are of similar value are most probable. We varied ω over the set $\{.1, .2, \ldots, 3.1\}$.

We have split the data in two and used the first half to estimate the kernel bandwidth through the following process. We first sample a true π, then we construct the KME and pick the bandwidth σ which minimizes $\|\pi - \hat{\pi}\|_{\ell_1}$. We performed the search on σ by using Matlab's function *fminbnd*. For both the SKM and the full case we allowed for 100 iterations. We have used the Gaussian kernel to create the sparse KME of the ℓ^{th} distribution, with sparsity level of $k_\ell = \sqrt{n_\ell}$, where n_ℓ is the size of the available sample from distribution ℓ. Since the α coefficients depend on σ, and for each set $\mathcal{I}^{(\ell)}$ we perform a search over several values of σ, we did not compute α iteratively as we constructed $\mathcal{I}^{(\ell)}$. Instead, once the construction of $\mathcal{I}^{(\ell)}$ was finished, we used the preconditioned conjugate gradient method to obtain α.

Once σ was estimated, we then accessed the second half of the data to test the performance for both the SKM and the full KME for different values of ω. The results are shown in Fig. 5. We have also plotted for perspective a "blind" estimation of π, which uniformly at random picks a vector $\hat{\pi}$. A comparison of the computation times for the sparse KME and the full KME is shown in Table II, where we have averaged over all values of ω.

TABLE II
COMPUTATION TIMES FOR BOTH FULL AND SPARSE KME, AVERAGED OVER ALL VALUES OF ω

	σ estimation	$\hat{\pi}$ computation	Total
Full	110.4s	5.08s	1.9mins
SKM	5.9s	5.07s	11s

Notice from Table II that, in the SKM case, the estimation of σ takes about the same time as the computation of $\hat{\pi}$. This is due to the fact that the main bottleneck of the algorithm is the computation of the set $\mathcal{I}$ which is independent of σ. It is in the estimation of σ that the full kernel mean is many times slower than SKM, as seen in the Table, SKM is about 10 times faster for the whole process.

C. Mean-Shift Clustering

We have based this experiment on the mean-shift algorithm as described in [53]. This algorithm is used in several image processing tasks and we will use it in the context of image segmentation. The goal is to form a clustering of the image pixels into different segments.

Each pixel is represented by a 5-dimensional vector (3 dimensions to describe color, and 2 for the position in the image), and the distribution of these feature vectors is estimated by the KDE. Denote the image pixels as $\{x_i\}_{i=1}^n$, $x_i \in \mathbb{R}^5$. The mean-shift algorithm shifts each point lying on the surface of the density closer to its closest peak (mode). Given a starting point x, the algorithm iteratively shifts x closer to its mode until the magnitude of the shift is smaller than some quantity γ. The shift exerted on x at each iteration requires the computation of the gradient of the KDE at the current position, making mean-shift computationally expensive. Denote the shifted points as $\{y_i\}_{i=1}^n$. Once all points are shifted close to the different modes, then any clustering algorithm can be performed to find the clusters. A clustering algorithm is described in [53], based on merging the modes' neighborhoods which are close. We used a code following these guidelines found at [54], slightly modified by increasing the distance used for modes' neighborhoods to merge.

In our experiments we used a 500×487 image of a painting by Piet Mondrian (*Composition A*), and compared our algorithm with the full density estimation case. We chose k_{max} to be the largest integer smaller than $\sqrt{n}$ and we have used the method for auto-selecting k_0 outlined in Section V-B, with $\epsilon = 10^{-8}$. We have used the Gaussian kernel and set the bandwidth according to Equation (18) in [55], which is specifically suggested for mode-based clustering. We compare the SKM approach to a method based on Locality Sensitive Hashing (LSH, see [56], [57]). This method finds for each point and with high probability its nearest neighbors. It then approximates the KDE locally by only using the effect from such neighbors. We chose 5 nearest neighbors and to implement LSH we used the Matlab version of LSH available at [58] (we have used the e2lsh scheme with three hash tables per picture). See [58], [59] for details on LSH.

TABLE III
TIME AND PERFORMANCE COMPARISON FOR MEAN SHIFT ALGORITHM

	Time			Performance	
	Preparation	Mean Shift	Total	$d_i(\cdot,\mathcal{B})$	$\widehat{d}_H(\cdot,\mathcal{B})$
$\mathcal{B}$	0	2.1hrs	2.1hrs	0	0
$\mathcal{A}_{\text{SKM}}$	15.65s	39.12s	54.77s	0.006	0.015
$\mathcal{A}_{\text{LSH}}$	12.7s	7.1mins	7.32mins	0.034	0.02

We present two indicators to evaluate the performance between the clustering resulting from the full KDE and that resulting from the approximate KDE. In the following, let $\mathcal{B}$ be used to indicate that the full kernel density estimate has been used, while $\mathcal{A}$ indicates either the SKM or the LSH approaches. With a slight abuse of notation, let $\mathcal{A}$ and $\mathcal{B}$ also indicate their resulting clusterings.

Discrepancy Index: Our first performance measure, which we call the discrepancy index d_i, is somehow intuitive, and it describes the ratio of the number of vectors x_ℓ that the approximate methods shifted by more than δ away from their full method counterpart. δ is here some tolerance threshold, which we have set to three times the kernel bandwidth. More precisely, if $\{x_\ell\}_{\ell=1}^n$ indicate the picture pixels and $y_\ell^{\mathcal{A}}$, $y_\ell^{\mathcal{B}}$ are the shifted versions of x_ℓ according to density estimation methods $\mathcal{A}$ and $\mathcal{B}$ respectively, then

$$d_i(\mathcal{A},\mathcal{B}) = \frac{1}{n}\sum_\ell \mathbf{1}_{\{\|y_\ell^{\mathcal{A}}-y_\ell^{\mathcal{B}}\|>\delta\}}.$$

Hausdorff Distance: The second performance measure, which describes the Hausdorff distance between clusterings, was obtained from [60] and is denoted by d_H. To define the Hausdorff distance, let P be a distribution on $\mathbb{R}^d$ (in our case, P is the distribution of the image pixels on $\mathbb{R}^5$). Furthermore, let $\mathcal{X}$ be the set of subsets of $\mathbb{R}^d$ such that the distance between two sets A and B is $\rho(A,B) := P(A\Delta B)$, where Δ is the symmetric difference (to be precise, we deal with equivalence classes, where two sets A and B are equivalent if $\rho(A,B)=0$). Notice $\mathcal{X}$ is a metric space. Let $\mathcal{B}\subset\mathcal{X}$, and define $\rho(A,\mathcal{B}) := \min_{B\in\mathcal{B}}\rho(A,B)$. We interpret a subset $\mathcal{A}$ of $\mathcal{X}$ as a clustering, and an element A in $\mathcal{X}$ as a cluster. The Hausdorff distance between two clusterings is

$$d_H(\mathcal{A},\mathcal{B}) = \max\left\{\max_{A\in\mathcal{A}}\ \rho(A,\mathcal{B}),\ \max_{B\in\mathcal{B}}\ \rho(B,\mathcal{A})\right\}.$$

In words, d_H measures the furthest distance between elements of $\mathcal{A}$ to the clustering $\mathcal{B}$ and elements of $\mathcal{B}$ to the clustering $\mathcal{A}$ (that is, d_H measures the less overlap between clusters of different clusterings, as measured by P). Since we don't have access to P, the empirical version of d_H proposed in [60] is obtained by replacing P by the empirical probability measure. Letting $\widehat{\rho}(A,\mathcal{B}) := \min_{B\in\mathcal{B}}\frac{1}{n}\sum_{i=1}^n \mathbf{1}_{\{A\Delta B\}}(x_i)$, we have

$$\widehat{d}_H(\mathcal{A},\mathcal{B}) = \max\left\{\max_{A\in\mathcal{A}}\ \widehat{\rho}(A,\mathcal{B}),\ \max_{B\in\mathcal{B}}\ \widehat{\rho}(B,\mathcal{A})\right\}.$$

We use this latter quantity to measure the SKM performance.

The results are presented in Table III. In the table $\mathcal{B}$ indicates the full kernel density estimate has been used, $\mathcal{A}_{\text{SKM}}$ indicates the k-center based algorithm and $\mathcal{A}_{\text{LSH}}$ the LSH method. Note that both the SKM and the LSH approach present significant computational advantages. The SKM approach, however, manages to be faster while incurring half the discrepancy of the LSH and about the same Hausdorff distance.

TABLE IV
TIME COMPLEXITY AND MEMORY COMPARISON AMONG SELECTED METHODS. n IS THE DATA SIZE, k THE APPROXIMATION SIZE, AND T THE NUMBER OF ITERATIONS FOR THE k-MEANS ALGORITHM FROM MATLAB

Method	Selection time	α computation time	Memory
SKM	nk	k^3	k^2
RAND	$k\log n$	k^3	k^2
L2	n^2	$\leq k^3$	k^2
KMEANS	nkT	k^3	k^2
HERD	n^2+k^3	(combined with selection)	k^2

TABLE V
SPARSITY LEVEL REQUIRED FOR AN ACCURACY OF 10^{-3}

	SKM	RAND	L2	KMEANS	HERD
banana	0.6443	0.8905	1	0.9068	1
image	0.7367	0.7933	1	0.8540	1
ringnorm	0.6603	0.6779	1	0.6787	.8325
breast-cancer	0.6768	0.6062	1	0.5905	1
heart	0.7455	0.8386	1	0.8384	1
thyroid	0.7257	0.9410	1	0.9383	1
diabetes	0.6405	0.7885	1	0.7859	1
german	0.5648	0.5900	1	0.5889	.8
twonorm	0.6202	0.5962	0.9863	0.6002	.7725
waveform	0.6557	0.6876	1	0.6843	1
iris	0.7667	0.8536	1	0.8782	1

D. Comparison with Other Subset Selection Strategies

To further illustrate the performance of SKM, we look at the sparsity required to achieve a given accuracy ϵ, that is, the smallest value k_0/k_{max} for which the quantity $\bar{E}_{|\mathcal{I}|} := \|\bar{z}-z_{\mathcal{I}}\|^2/\|\bar{z}\|^2$ is smaller than ϵ, where $k_0 = |\mathcal{I}|$, (see Section V-B). Note we selected $\bar{E}_{|\mathcal{I}|}$ as opposed to the term $E_{|\mathcal{I}|}$ used for autoselection because $\bar{E}_{|\mathcal{I}|}$ is more interpretable and for these experiments we are not interested in efficient autoselection. We have applied our method for 11 distinct benchmark datasets, listed in Table V. We present these sparsity values for $\epsilon = 10^{-3}$ and also the corresponding values for four other methods:1) RAND, which chooses the set $\mathcal{I}$ uniformly at random, 2) L2, which chooses $\mathcal{I}$ by sampling according to the squared norm of the columns of the kernel matrix (see [38]), 3) KMEANS which picks as the representative set not a subset of the data but the results from the k-means algorithm and constructs the kernel matrix according to these (see again [38]), and 4) HERD, which uses kernel herding (see [19]) to select $\mathcal{I}$ and $\alpha_{\mathcal{I}}$. Table IV shows the time and memory complexities for these algorithms. The Wilcoxon signed rank test was performed pairwise comparing the performance of SKM to the other methods. The p-values for SKM with respect to RAND, L2, KMEANS and HERD are respectively .024, .001, .019, and .001 favoring SKM. Note that RAND and KMEANS have similar performance, and in fact the p-value between the two is .64. We also present the complete error plot for the banana dataset in Figure 6. Note that in this case other approximations show an initial advantage because they are more likely to pick elements from dense areas, which

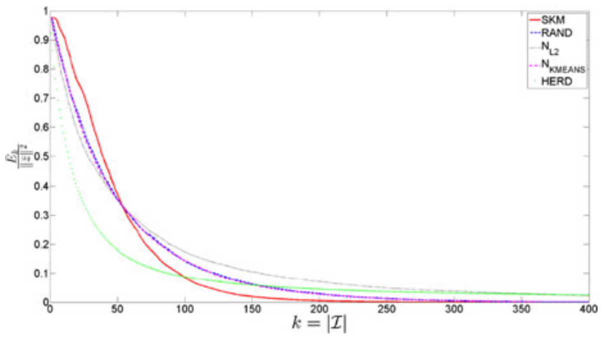

Fig. 6. Error comparison among different methods for the banana data set. SKM incurs a larger initial error but quickly decreases as it captures more of the fine structure of the distribution. Note that RAND and $\mathrm{N}_{\mathrm{KMEANS}}$ perform very similarly, this can also be seen in table V.

TABLE VI
VALUES OF $D(\bar{z}\|z_{\mathcal{I}})$ AND $D(z_{\mathcal{I}}\|\bar{z})$ FOR DIFFERENT DATA SETS

	$D(\bar{z}\|z_{\mathcal{I}})$		$D(z_{\mathcal{I}}\|\bar{z})$	
	RAND	SKM	RAND	SKM
banana	0.092597	0.001805	0.129183	0.001613
image	0.451205	0.041305	0.212585	0.061584
ringnorm	0.003983	0.031736	0.009253	0.02853
breast-cancer	0.358253	0.002546	0.345895	4.56E-05
heart	0.001918	6.35E-16	0.005228	2.91E-16
thyroid	0.177317	0.000594	0.034616	0.000289
diabetes	0.031366	0.005474	0.014635	0.000102
german	0.008711	0.003855	0.008742	0.00203
twonorm	0.000131	0.000243	4.59E-05	0.000372
waveform	0.011473	0.000177	0.015064	0.000404
iris	0.043924	0.000395	0.022519	0.000104

for small values of $|\mathcal{I}|$ represents better the full distribution. However, as the size of $\mathcal{I}$ increases, the fine structure (e.g., the distribution tails) is better captured by SKM, since the k-center algorithm picks points far apart from each other. Note that Table V shows that SKM achieves better sparsity for a given accuracy. Equivalently, we can say SKM achieves better accuracy for a fixed sparsity level. In the appendix, we also report the time required to achieve an accuracy of 10^{-3} and find a similar advantage for SKM.

The sparse approximation strategy proposed in this paper can be a valid density if the α_i's are set to satisfy $\alpha_i \geq 0$ and $\sum_i \alpha_i = 1$. Therefore, we also evaluate the performance of the proposed sparse approximation according to the KL divergence, a common metric between distributions whose arguments must be density functions.

For the same benchmark data sets listed above, we computed the KL divergences $D(\bar{z}\|z_{\mathcal{I}})$ and $D(z_{\mathcal{I}}\|\bar{z})$ between the sparse and the full kernel mean. We used the auto-selection scheme proposed in Section V-B with $\epsilon = 10^{-7}$, and projected the resulting α onto the simplex to ensure we have a valid probability distribution. We have chosen a Gaussian kernel and used the Jaakkola heuristic [61] to compute the bandwidth. We compare the performance of SKM to that of RAND, which, as seen above, is similar to that of KMEANS. We have performed the Wilcoxon signed rank test [62] to determine if there is a significant advantage of the SKM. The test for both the case $D(\bar{z}\|z_{\mathcal{I}})$ and the case $D(z_{\mathcal{I}}\|\bar{z})$ yields a p-value of 0.0186, favoring the SKM method. The results are shown in Table VI. To understand this, note that in the extreme case in which one density is zero in a particular region while the other is positive, the KL divergence is infinite, so the KL divergence highly penalizes very low density approximations to positive density regions. The SKM accurately approximates low density regions since it selects outliers, while the random selection approach picks mostly points in populated regions.

E. Other Results

We have performed additional experiments that explore the performance of SKM as dimension increases. These results have been placed in the appendix, which is available as a supplemental file. In conclusion, the results suggest that the sparsity required to achieve a given accuracy increases as a function of dimension and decreases as a function of bandwidth. We have also compared the performance of SKM for the Laplacian and the Student-t kernels. In general they both exhibit a similar performance as for the Gaussian kernel, in terms of relative error $\bar{E}$. For the Euclidean embedding and class proportion estimation experiments, however, it is harder to set an effective bandwidth for the Student-t kernel.

VII. CONCLUSION

We have provided a method to rapidly and accurately build a sparse approximation of a kernel mean. We derived an incoherence based bound on the approximation error and recognized that, for abroad class of kernels, including translation invariant kernels, its minimization is equivalent to solving the k-center problem either on the feature space or the Euclidean space where the data lies. If desired, our construction of the sparse kernel mean may be slightly modified to provide a valid density function, which is important in some applications. Hence, the algorithm is versatile in that it works for both kinds of kernel means: the KDE and the KME. Our method also naturally lends itself to a sparsity auto-selection scheme.

We showed its computational advantages and its performance qualities in three specific applications. First, Euclidean embedding of distributions (for both KDE and KME), in which, for the KDE case, a valid density is needed to compute the KL divergence. Second, class proportion estimation (for the KME), which presents the amortization advantages of the SKM approach, in this case with respect to the bandwidth σ. Finally, mean-shift clustering (for the KDE), in which with less computation time than the LSH-based approach, it performs better with respect to the discrepancy index and similar with respect to the Hausdorff distance. In most instances the proposed sparse kernel mean method has shown to be orders of magnitude faster than the approach based on the full kernel mean. Furthermore, we compared its performance in terms of error, sparsity, and time with respect to four other subset selection schemes for several benchmark datasets. We find that, with statistical significance, SKM outperforms these methods. Finally, we also observed its performance in different settings, concerning dimension and kernel variability. These latter results are presented in detail in the appendix.

Acknowledgment

The authors thank L. Stoolman of the University of Michigan Department of Pathology for providing the de-identified flow cytometry dataset.

References

[1] E. Cruz Cortés and C. Scott, "Scalable sparse approximation of a sample mean," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 5274–5278.

[2] E. Cruz Cortés and C. Scott, SKM MATLAB code. 2015. [Online]. Available: http://web.eecs.umich.edu/ cscott/code.html#skm, Accessed on: Feb. 2015.

[3] D. M. Titterington *et al.*, "Comparison of discrimination techniques applied to a complex data set of head injured patients," *J. Roy. Statist. Soc. Series A (General)*, 1981, pp. 145–175.

[4] D. J. Hand, "A comparison of two methods of discriminant analysis applied to binary data," *Biometrics*, vol. 39, pp. 683–694, 1983.

[5] M. J. Desforges, P. J. Jacob, and J. E. Cooper, "Applications of probability density estimation to the detection of abnormal conditions in engineering," *Proc. Institution Mech. Eng., Part C, J. Mech. Eng. Sci.*, vol. 212, no. 8, pp. 687–703, 1998.

[6] D. Yeung and C. Chow, "Parzen-window network intrusion detectors," in *Proc. 16th Int. Conf. Pattern Recog.*, 2002, vol. 4, pp. 385–388.

[7] M. Markou and S. Singh, "Novelty detection: A review—Part 1: Statistical approaches," *Signal Process.*, vol. 83, no. 12, pp. 2481–2497, 2003.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surveys*, vol. 41, no. 3, p. 15, 2009.

[9] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[10] I. Steinwart and A. Christmann, *Support Vector Machines*. New York, NY, USA: Springer, 2008.

[11] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A Hilbert space embedding for distributions," in *Algorithmic Learning Theory*. New York, NY, USA: Springer, 2007, pp. 13–31.

[12] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 723–773, 2012.

[13] K. Fukumizu, L. Song, and A. Gretton, "Kernel Bayes' rule," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1737–1745.

[14] P. Gurram and H. Kwon, "Contextual SVM for hyperspectral classification using Hilbert space embedding," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2012, pp. 5470–5473.

[15] C. Scovel, D. Hush, I. Steinwart, and J. Theiler, "Radial kernels and their reproducing kernel hilbert spaces," *J. Complexity*, vol. 26, no. 6, pp. 641–660, 2010.

[16] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[17] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[18] Y. Chen, M. Welling, and A. Smola, "Super-samples from kernel herding," in *Proc. Conf. Uncertainty Artif. Intell.*, 2010.

[19] F. Bach, S. Lacoste-Julien, and G. Obozinski, "On the equivalence between herding and conditional gradient algorithms," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012.

[20] B. Jeon and D. A. Landgrebe, "Fast Parzen density estimation using clustering-based branch and bound," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 9, pp. 950–954, Sep. 1994.

[21] M. Girolami and C. He, "Probability density estimation from optimally condensed data samples," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1253–1264, Oct. 2003.

[22] S. Chen, X. Hong, and C. J. Harris, "An orthogonal forward regression technique for sparse kernel density estimation," *Neurocomputing*, vol. 71, no. 4, pp. 931–943, 2008.

[23] M. Schafföner, E. Andelic, M. Katz, S. E. Krüger, and A. Wendemuth, "Memory-efficient orthogonal least squares kernel density estimation using enhanced empirical cumulative distribution functions," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2007, pp. 428–435.

[24] D. W. Scott and W. F. Szewczyk, "From kernels to mixtures," *Technometrics*, vol. 43, no. 3, pp. 323–335, 2001.

[25] A. R. Runnalls, "Kullback-Leibler approach to Gaussian mixture reduction," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 3, pp. 989–999, Jul. 2007.

[26] D. Schieferdecker and M. F. Huber, "Gaussian mixture reduction via clustering," in *Proc. 12th Int. Conf. Inf. Fusion*, 2009, pp. 1536–1543.

[27] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 381–396, Mar. 2002.

[28] P. Bruneau, M. Gelgon, and F. Picarougne, "Parsimonious reduction of Gaussian mixture models with a variational-Bayes approach," *Pattern Recognit.*, vol. 43, no. 3, pp. 850–858, 2010.

[29] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *J. Comput. Phys.*, vol. 73, no. 2, pp. 325–348, 1987.

[30] A. G. Gray and A. W. Moore, "N-body problems in statistical learning," in *Proc. Adv. Neural Inf. Process. Syst. 13*, vol. 4, 2000, pp. 521–527.

[31] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis, "Improved fast Gauss transform and efficient kernel density estimation," in *Proc. 9th IEEE Int. Conf. Comput. Vision*, 2003, pp. 664–671.

[32] D. Lee, A. Gray, and A. W. Moore, "Dual-tree fast Gauss transforms," in *Proc. Adv. Neural Inf. Process. Syst. 18 (Dec. 2005)*, 2006.

[33] Y. Zheng, J. Jestes, J. M. Phillips, and F. Li, "Quality and efficiency for kernel density estimates in large data," in *Proc. Int. Conf. Manage. Data*, 2013, pp. 433–444.

[34] J. M. Phillips, "ε-samples for kernels," in *Proc. 24th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2013, pp. 1622–1632.

[35] S. Joshi, R. V. Kommaraji, J. M. Phillips, and S. Venkatasubramanian, "Comparing distributions and shapes using the kernel distance," in *Proc. 27th Annu. Symp. Comput. Geometry*, 2011, pp. 47–56.

[36] J. Zhao and D. Meng, "FastMMD: Ensemble of circular discrepancy for efficient two-sample test," *Neural Comput.*, vol. 27, no. 6, pp. 1345–1372, 2015.

[37] P. Drineas and M. W. Mahoney, "On the Nyström method for approximating a gram matrix for improved kernel-based learning," *J. Mach. Learn. Res.*, vol. 6, pp. 2153–2175, 2005.

[38] S. Kumar, M. Mohri, and A. Talwalkar, "Sampling methods for the Nyström method," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 981–1006, 2012.

[39] W. B. March and G. Biros, "Far-field compression for fast kernel summation methods in high dimensions," *Appl. Comput. Harmonic Anal.*, 2015, in press.

[40] Z. Noumir, P. Honeine, and C. Richard, "One-class machines based on the coherence criterion," in *Proc. IEEE Statist. Signal Process. Workshop*, 2012, pp. 600–603.

[41] V. V. Vazirani, *Approximation Algorithms*. New York, NY, USA: Springer, 2001.

[42] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoretical Comput. Sci.*, vol. 38, pp. 293–306, 1985.

[43] D. S. Hochbaum, *Approximation Algorithms for NP-Hard Problems*. Boston, MA, USA: PWS-Kent, 1996.

[44] B. van Rooyen, A. K. Menon, and R. C. Williamson, "An average classification algorithm," *ArXiv e-prints*, arXiv:1506.01520, 2015.

[45] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 272–279.

[46] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.

[47] W. G. Finn, K. M. Carter, R. Raich, L. M. Stoolman, and A. O. Hero, "Analysis of clinical flow cytometric immunophenotyping data by clustering on statistical manifolds: Treating flow cytometry data as high-dimensional objects," *Cytometry Part B, Clinical Cytometry*, vol. 76, no. 1, pp. 1–7, 2009.

[48] P. Hall, "On the non-parametric estimation of mixture proportions," *J. Roy. Statist. Soc. Series B (Methodol.)*, vol. 43, pp. 147–156, 1981.

[49] D. M. Titterington, "Minimum distance non-parametric estimation of mixture proportions," *J. Roy. Statist. Soc.*, vol. 45, no. 1, pp. 37–46, 1983.

[50] T. Sanderson and C. Scott, "Class proportion estimation with application to multiclass anomaly rejection," in *Proc. 17th Int. Conf. Artif. Intell. Statist.*, 2014.

[51] Y. LeCun, The mnist database. [Online]. Available: http://yann.lecun.com/exdb/mnist, accessed on: Sep. 24, 2014.

[52] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[53] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[54] B. Finkston, Mean shift clustering, 2014. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/10161-mean-shift-clusteri-ng, Accessed on: Sep. 24, 2014.
[55] Y. Chen, C. R. Genovese, and L. Wasserman, "A comprehensive approach to mode clustering," *Eletron. J. Statist.*, vol. 10, no. 1, pp. 210–241, 2016.
[56] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. 25th Int. Conf. Very Large Data Bases*, 1999, vol. 99, pp. 518–529.
[57] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, 2006, pp. 459–468.
[58] G. Shakhnarovich, Locality Sensitive Hashing, 2014. [Online]. Available: http://ttic.uchicago.edu/ gregory, accessed on: Sep. 24, 2014.
[59] A. Andoni, LSH algorithm and implementation, 2014. [Online]. Available: http://www.mit.edu/ andoni/LSH, accessed on: Sep. 24, 2014.
[60] J. E. Chacón, "A population background for nonparametric density-based clustering," arXiv:1408.1381, 2014.
[61] T. Jaakkola, M. Diekhans, and D. Haussler, "Using the fisher kernel method to detect remote protein homologies," in *Proc. 7th Int. Conf. Intell. Syst. Molecular Biol.*, 1999, vol. 99, pp. 149–158.
[62] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.

Efrén Cruz Cortés is a Ph.D. candidate in Electrical Engineering at the University of Michigan. The focus of his research is machine learning.

Clayton Scott (M'14) is an associate professor of EECS and of Statistics at the University of Michigan. He received his A.B. in Mathematics from Harvard in 1998, and his M.S. and Ph.D. in Electrical Engineering from Rice in 2000 and 2004. In 2011 he received the NSF CAREER Award. His research interests include statistical learning theory, algorithms, and applications.